EXHIBIT A

Part 1 of 3

US007596784B2

(12) **United States Patent**
Abrams et al.

(10) **Patent No.: US 7,596,784 B2**
(45) **Date of Patent: Sep. 29, 2009**

(54) **METHOD SYSTEM AND APPARATUS FOR PROVIDING PAY-PER-USE DISTRIBUTED COMPUTING RESOURCES**

(75) Inventors: **Peter C. Abrams**, Belmont, CA (US); **Rajeev Bharadhwaj**, Los Altos, CA (US); **Swami Nathan**, San Jose, CA (US); **Robert Rodriguez**, Fremont, CA (US); **Craig W. Martyn**, San Francisco, CA (US)

(73) Assignee: **Symantec Operating Corporation**, Cupertino, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 1665 days.

(21) Appl. No.: **09/950,559**

(22) Filed: **Sep. 10, 2001**

(65) **Prior Publication Data**

US 2002/0166117 A1 Nov. 7, 2002

Related U.S. Application Data

(60) Provisional application No. 60/232,052, filed on Sep. 12, 2000.

(51) **Int. Cl.**
G06F 9/44 (2006.01)
G06F 9/445 (2006.01)
G06F 15/173 (2006.01)

(52) **U.S. Cl.** **717/172**; 717/120; 717/127; 717/178; 709/224; 709/225; 709/226

(58) **Field of Classification Search** 717/168–178; 709/201–244
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,732,275 A * 3/1998 Kullick et al. 717/170
5,765,205 A 6/1998 Breslau et al.
5,903,762 A * 5/1999 Sakamoto et al. 717/178
6,065,123 A 5/2000 Chou et al.
6,771,290 B1 * 8/2004 Hoyle 715/745
2002/0178244 A1 * 11/2002 Brittenham et al. 709/223
2003/0200541 A1 * 10/2003 Cheng et al.

OTHER PUBLICATIONS

"Security Roadmap for Ejasent's Utility Computing Network", 2001, 18 pgs.

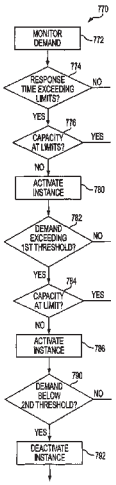
* cited by examiner

Primary Examiner—Chuck O Kendall
(74) *Attorney, Agent, or Firm*—Meyertons Hood Kivlin Kowert & Goetzel, P.C.

(57) **ABSTRACT**

Method, system, apparatus, and computer program and computer program product provide on-demand, scalable computational resources to application providers over a distributed network and system. Resources are made available based on demand for applications. Application providers are charged fees based on the amount of resources utilized to satisfy the needs of the application. In providing compute resources, method and apparatus is capable of rapidly activating a plurality of instances of the applications as demand increases and to halt instances as demand drops. Application providers are charged based on metered amount of computational resources utilized in processing their applications. Application providers access the network to distribute applications onto network to utilize distributed compute resources for processing of the applications. Application providers are further capable of monitoring, updating and replacing distributed applications. Apparatus and system includes plurality of computing resources distributed across a network capable of restoring and snapshotting provisioned applications based on demand.

25 Claims, 25 Drawing Sheets



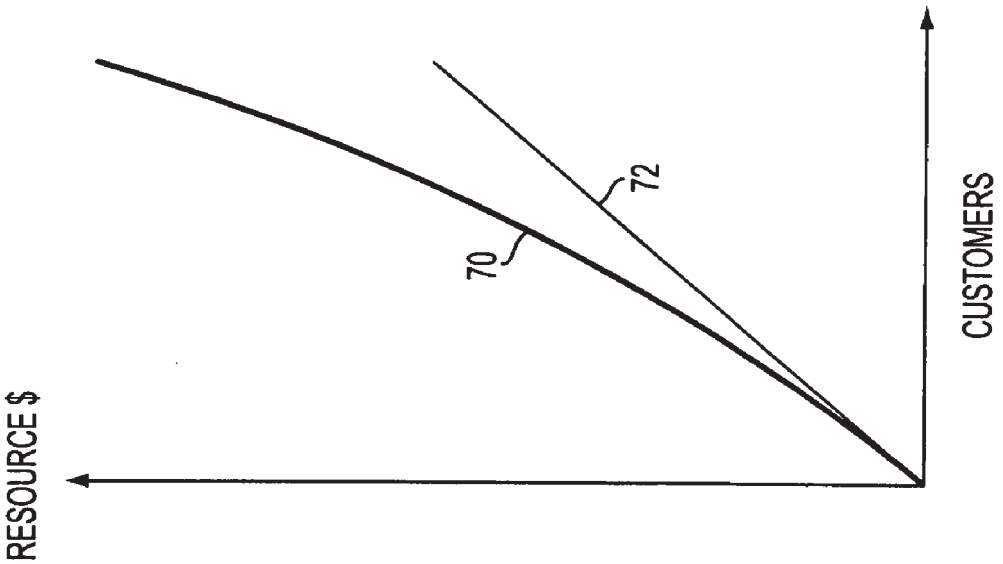

RESOURCE $
CUSTOMERS
70
72

FIG. 2

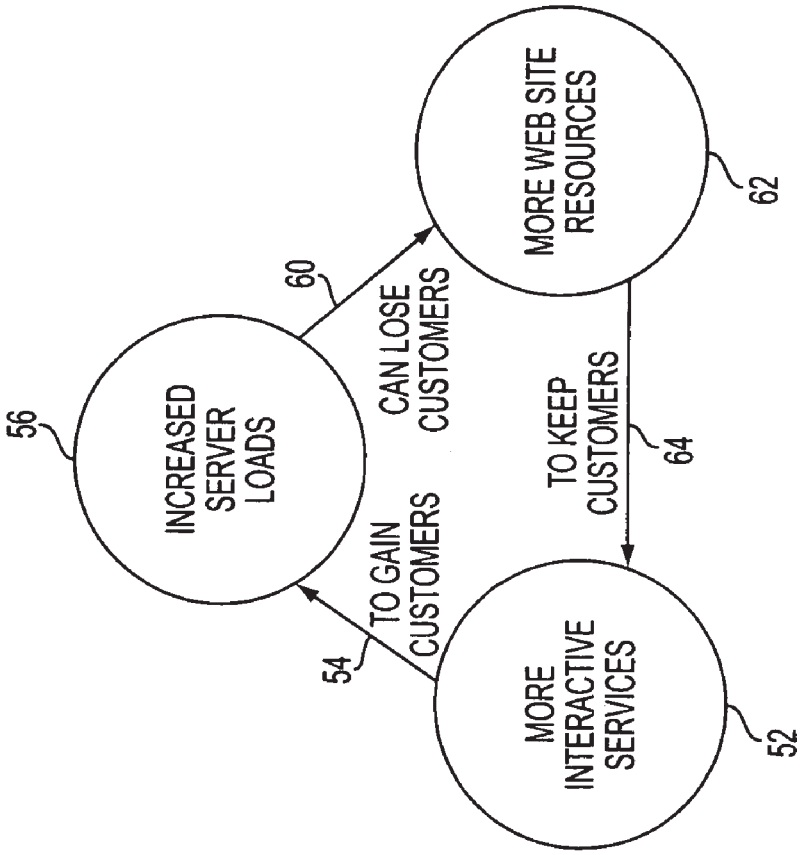

INCREASED SERVER LOADS
MORE WEB SITE RESOURCES
MORE INTERACTIVE SERVICES
CAN LOSE CUSTOMERS
TO GAIN CUSTOMERS
TO KEEP CUSTOMERS
56
60
62
54
64
52

FIG. 1

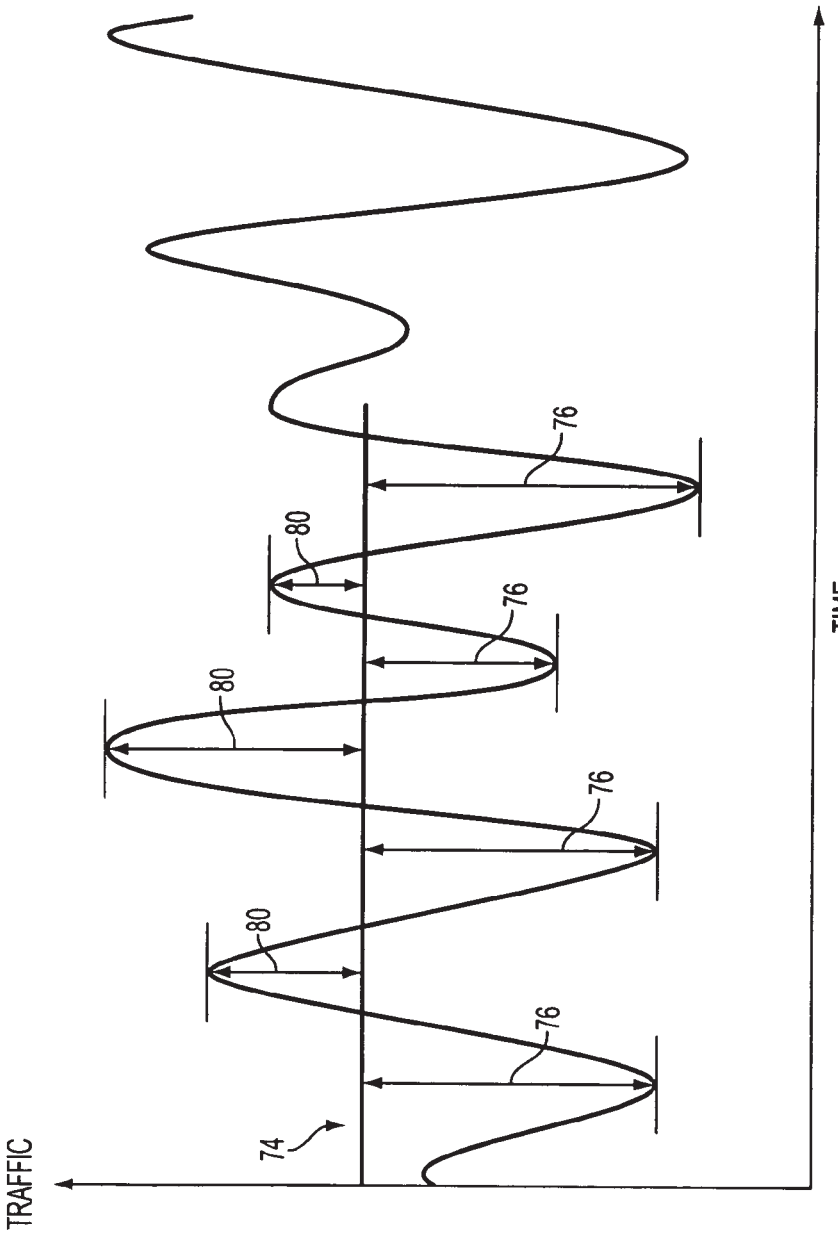


FIG. 3

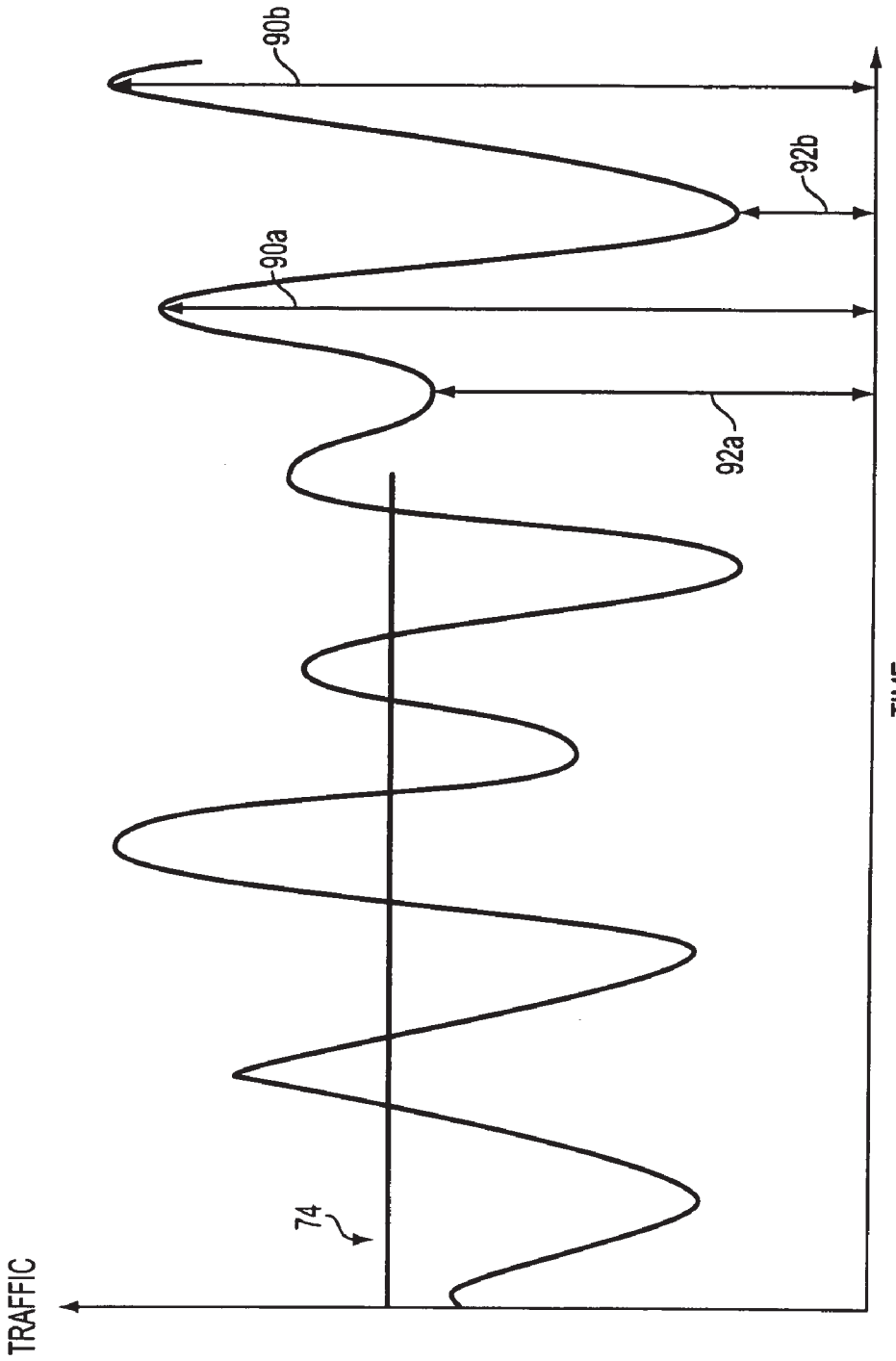


FIG. 4

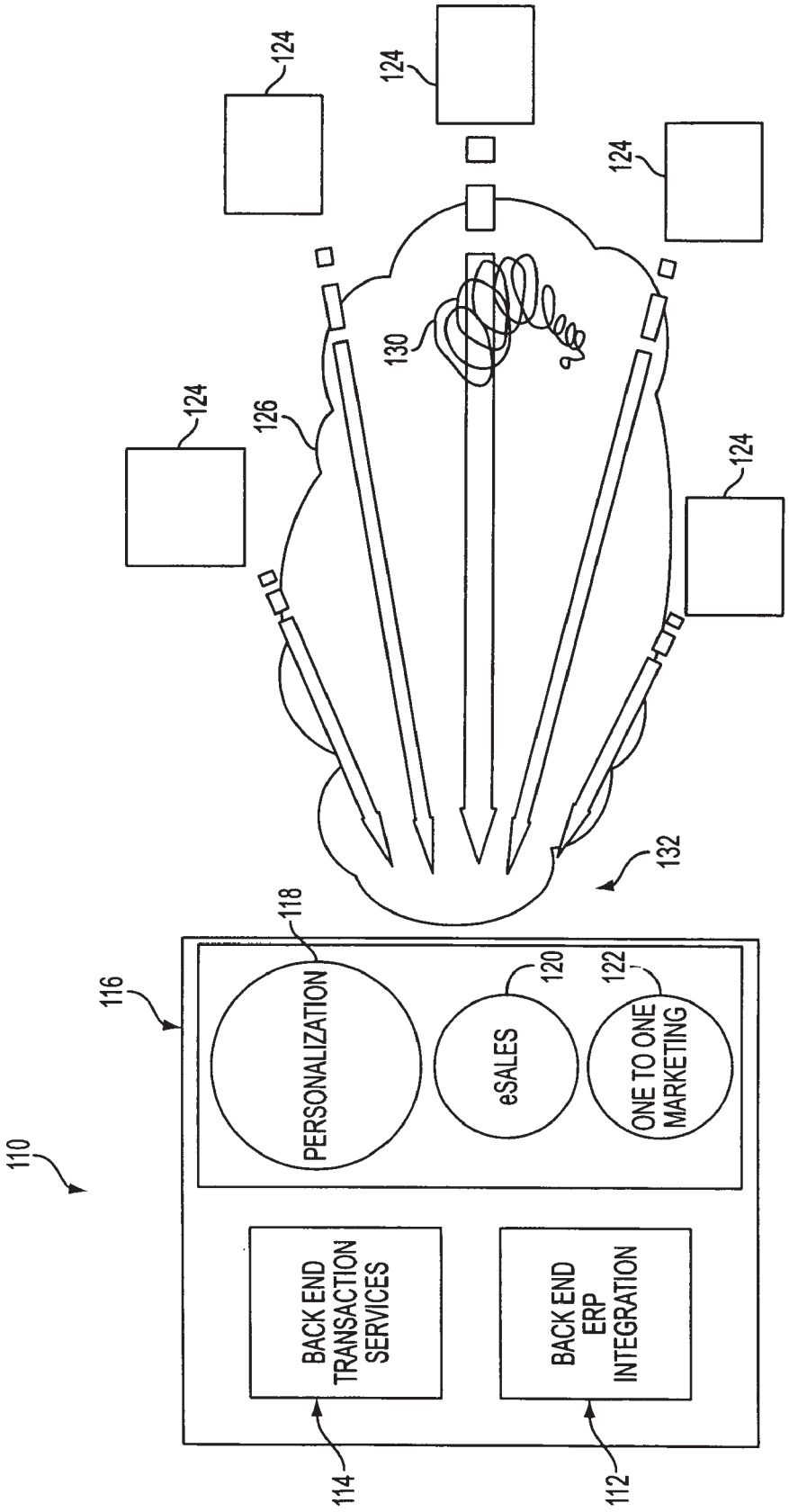

124
124
124
124
124
130
126
132
118
116
110
PERSONALIZATION
120
eSALES
122
ONE TO ONE MARKETING
BACK END TRANSACTION SERVICES
BACK END ERP INTEGRATION
114
112

FIG. 5

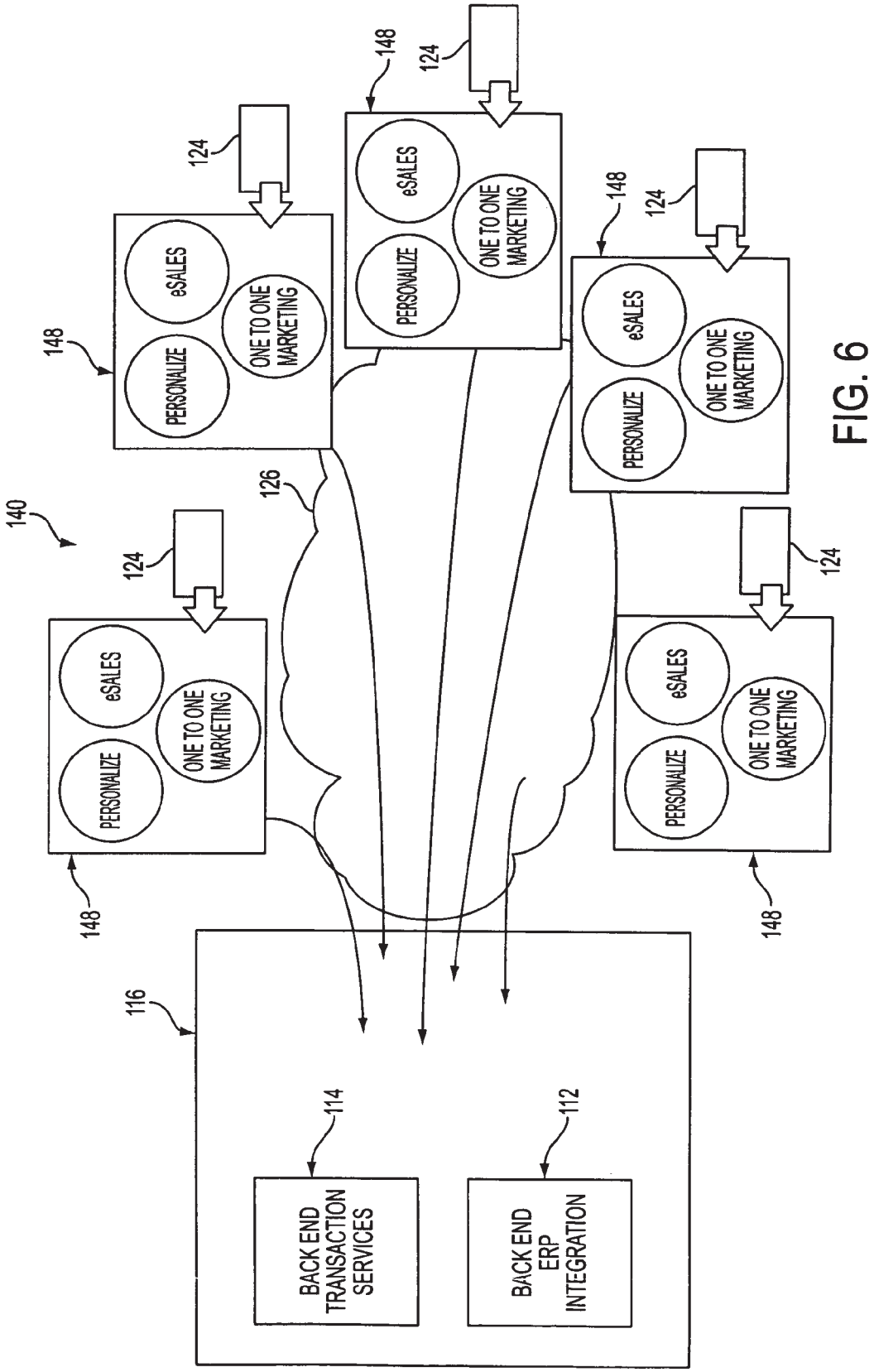

140
126
116
114
112
BACK END TRANSACTION SERVICES
BACK END ERP INTEGRATION
124
148
eSALES
ONE TO ONE MARKETING
PERSONALIZE

FIG. 6

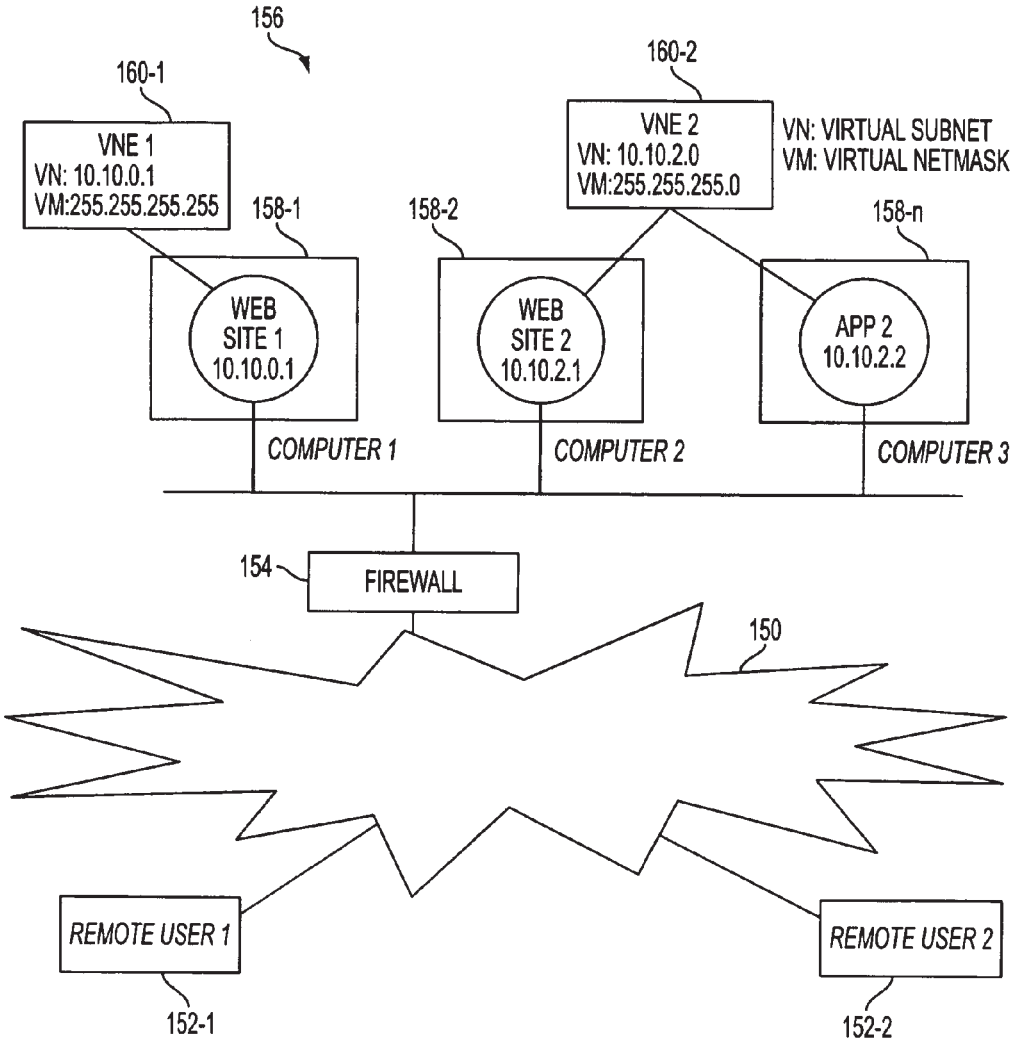

156
160-1
VNE 1
VN: 10.10.0.1
VM:255.255.255.255
160-2
VNE 2
VN: 10.10.2.0
VM:255.255.255.0
VN: VIRTUAL SUBNET
VM: VIRTUAL NETMASK
158-1
158-2
158-n
WEB SITE 1 10.10.0.1
WEB SITE 2 10.10.2.1
APP 2 10.10.2.2
COMPUTER 1
COMPUTER 2
COMPUTER 3
154
FIREWALL
150
REMOTE USER 1
REMOTE USER 2
152-1
152-2

FIG. 7

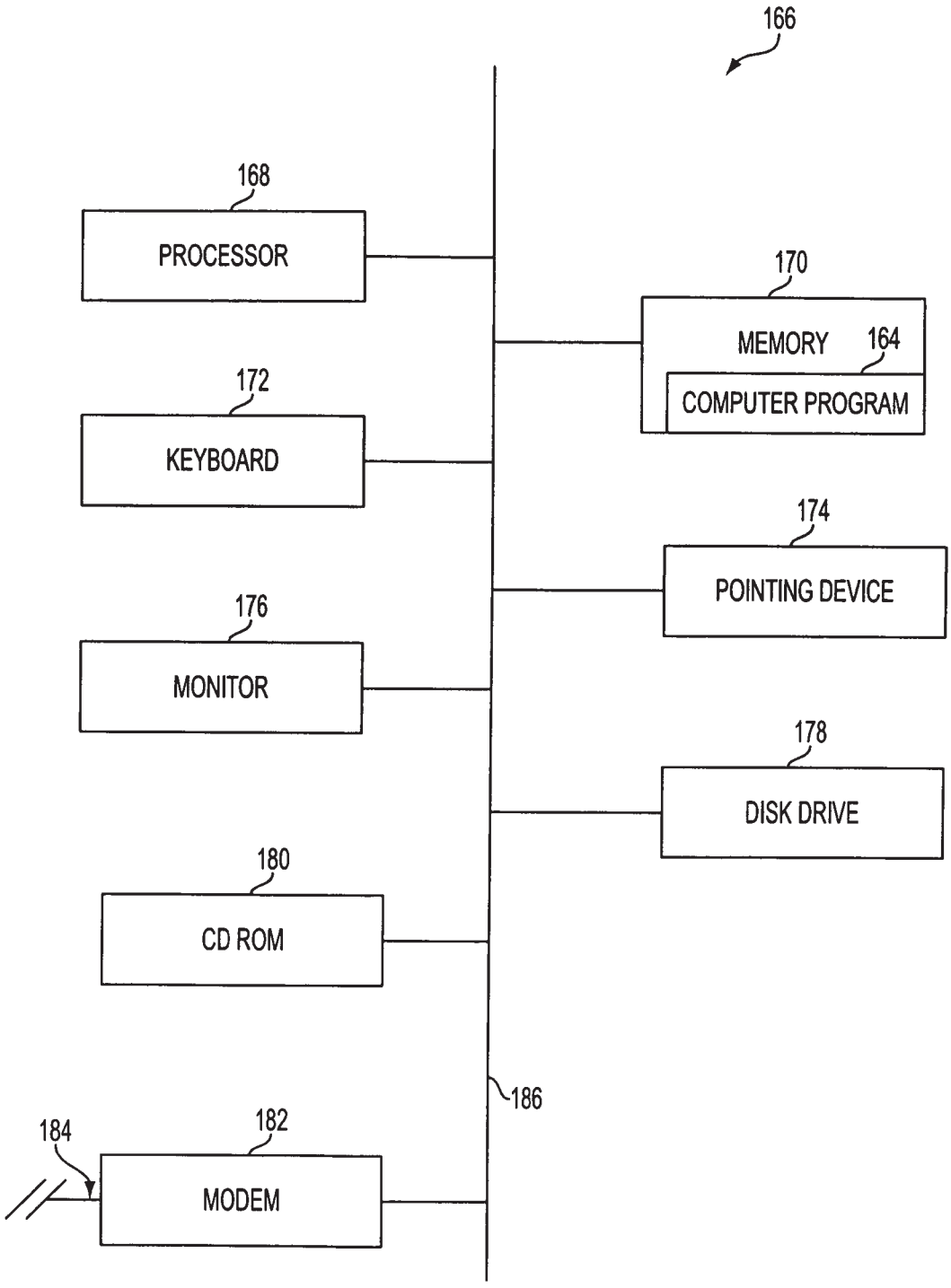

166
168
PROCESSOR
170
MEMORY
164
COMPUTER PROGRAM
172
KEYBOARD
174
POINTING DEVICE
176
MONITOR
178
DISK DRIVE
180
CD ROM
186
184
182
MODEM

FIG. 8

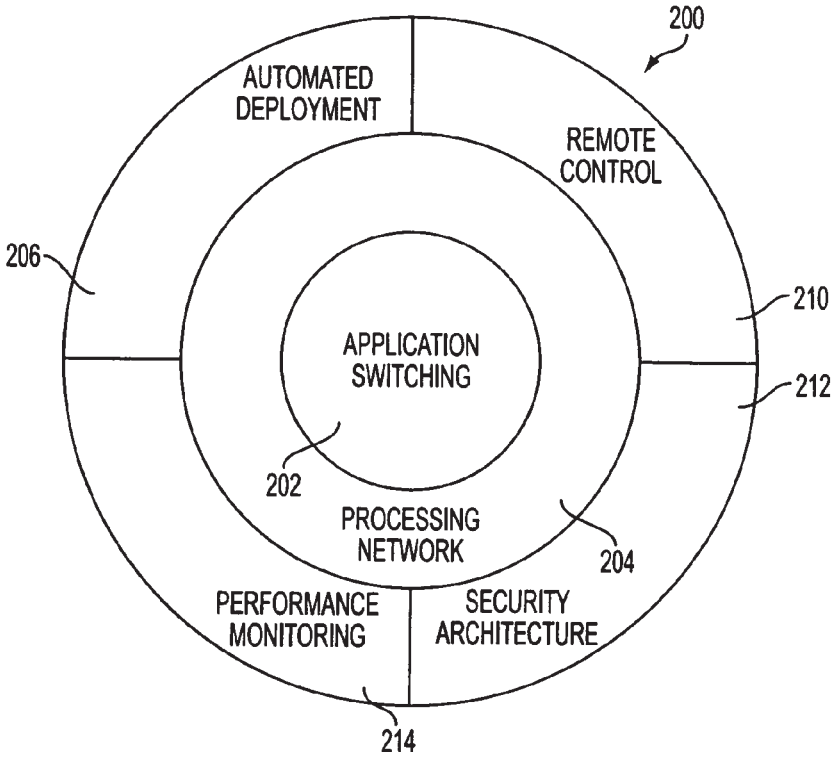

200
AUTOMATED DEPLOYMENT
REMOTE CONTROL
206
210
APPLICATION SWITCHING
212
202
PROCESSING NETWORK
204
PERFORMANCE MONITORING
SECURITY ARCHITECTURE
214

FIG. 9

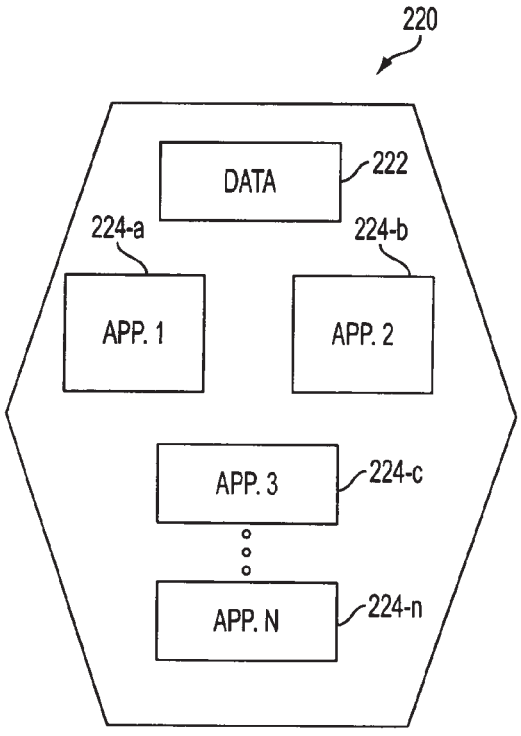

220
DATA
222
224-a
224-b
APP. 1
APP. 2
APP. 3
224-c
APP. N
224-n

FIG. 10

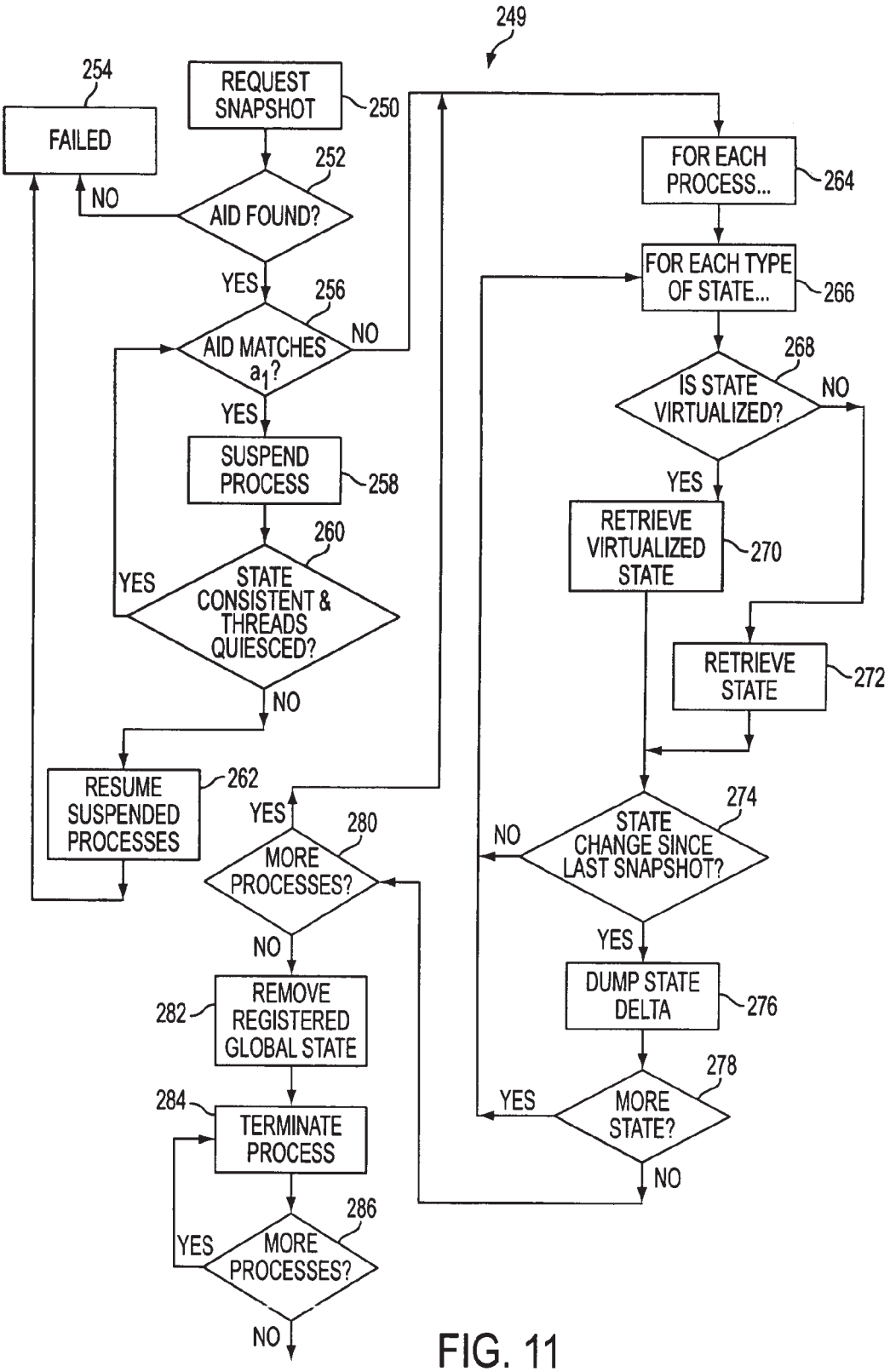

249
254
FAILED
REQUEST SNAPSHOT
250
NO
252
AID FOUND?
YES
256
AID MATCHES a_1?
NO
YES
SUSPEND PROCESS
258
260
YES
STATE CONSISTENT & THREADS QUIESCED?
NO
RESUME SUSPENDED PROCESSES
262
YES
280
MORE PROCESSES?
NO
282
REMOVE REGISTERED GLOBAL STATE
284
TERMINATE PROCESS
286
YES
MORE PROCESSES?
NO
FOR EACH PROCESS...
264
FOR EACH TYPE OF STATE...
266
268
IS STATE VIRTUALIZED?
NO
YES
RETRIEVE VIRTUALIZED STATE
270
RETRIEVE STATE
272
274
NO
STATE CHANGE SINCE LAST SNAPSHOT?
YES
DUMP STATE DELTA
276
278
YES
MORE STATE?
NO

FIG. 11

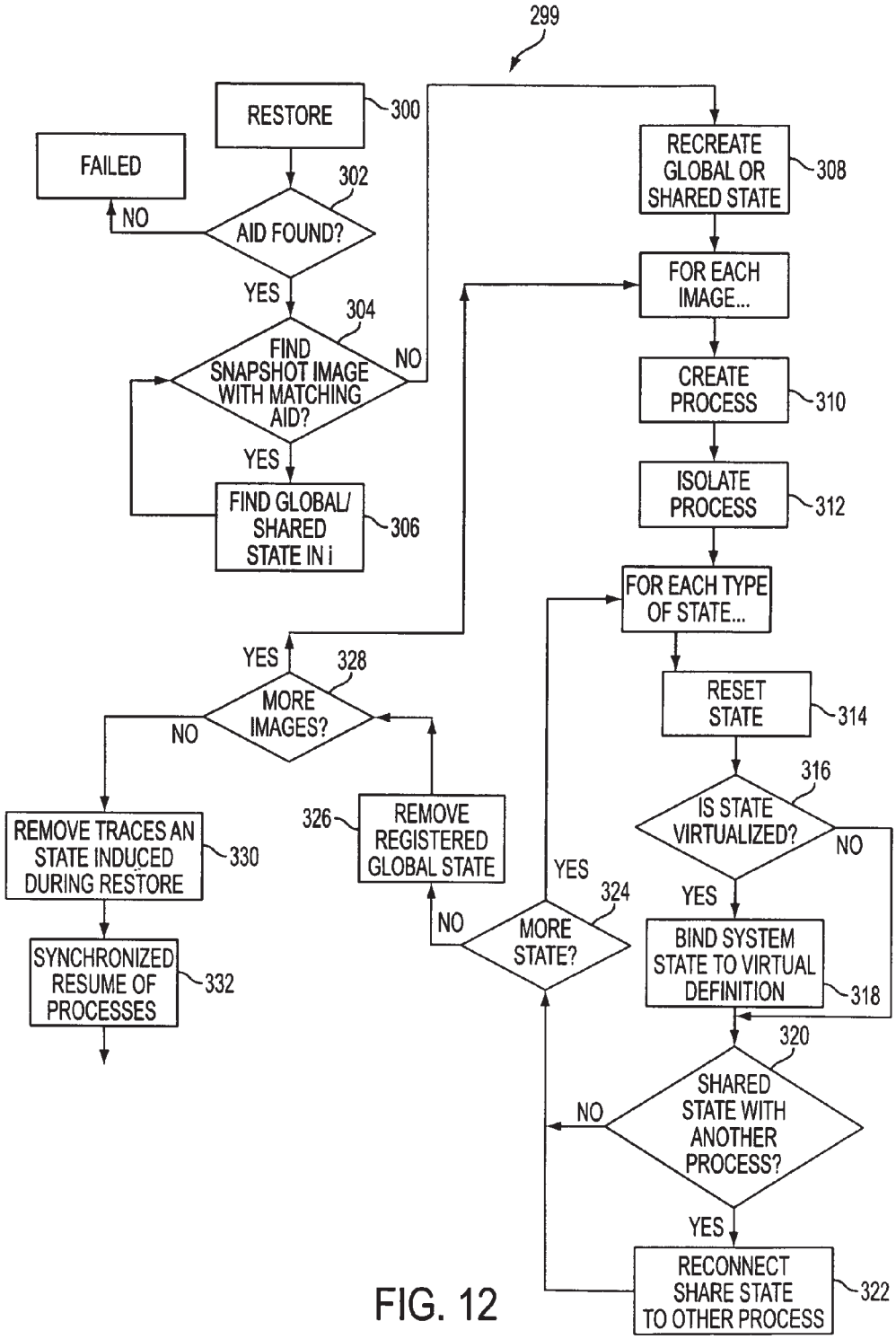

299
RESTORE
300
FAILED
NO
302
AID FOUND?
YES
304
FIND SNAPSHOT IMAGE WITH MATCHING AID?
NO
YES
FIND GLOBAL/ SHARED STATE IN i
306
RECREATE GLOBAL OR SHARED STATE
308
FOR EACH IMAGE...
CREATE PROCESS
310
ISOLATE PROCESS
312
FOR EACH TYPE OF STATE...
RESET STATE
314
316
IS STATE VIRTUALIZED?
NO
YES
BIND SYSTEM STATE TO VIRTUAL DEFINITION
318
320
SHARED STATE WITH ANOTHER PROCESS?
NO
YES
RECONNECT SHARE STATE TO OTHER PROCESS
322
YES
324
MORE STATE?
NO
326
REMOVE REGISTERED GLOBAL STATE
YES
328
MORE IMAGES?
NO
REMOVE TRACES AN STATE INDUCED DURING RESTORE
330
SYNCHRONIZED RESUME OF PROCESSES
332

FIG. 12

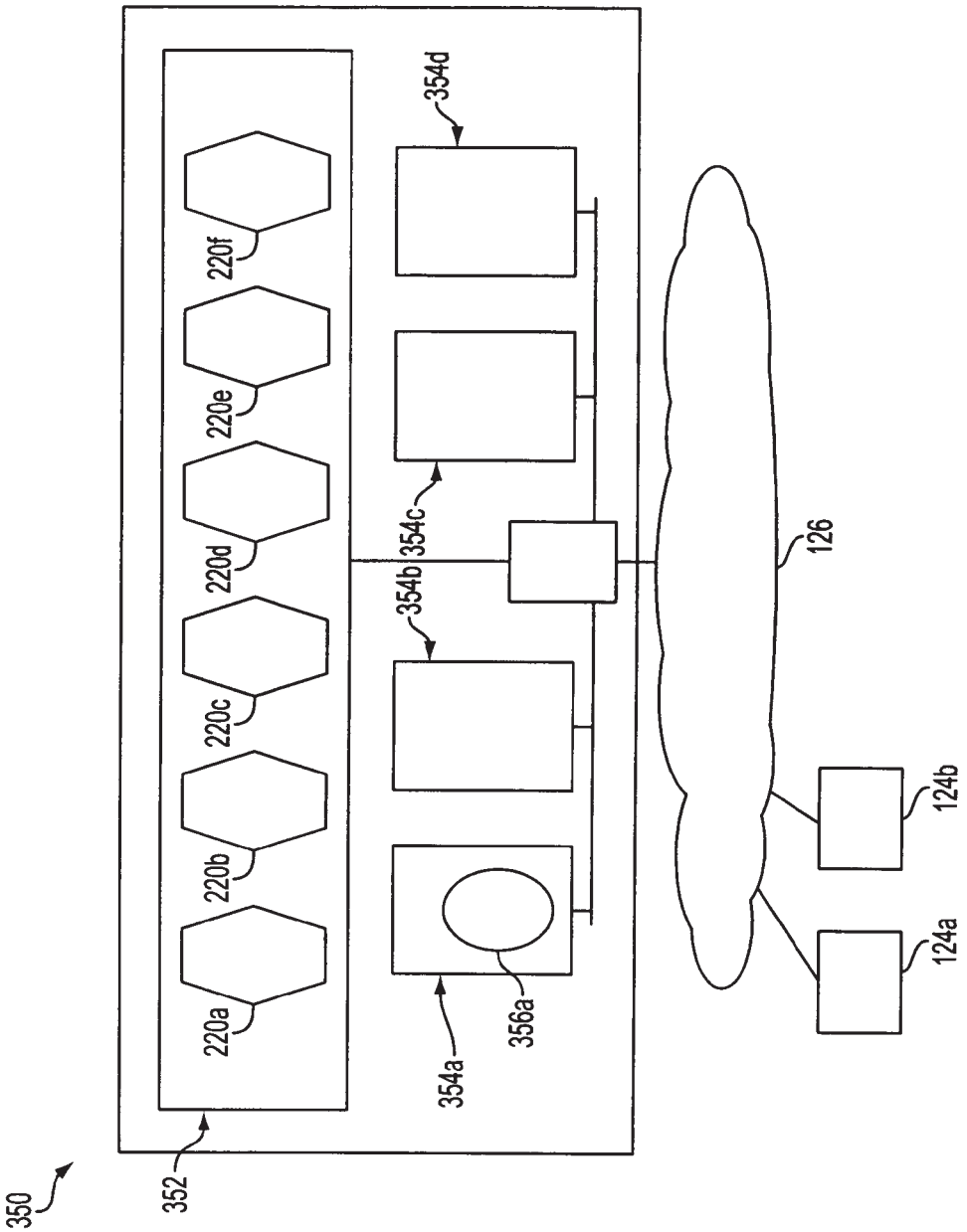

350
352
220a
220b
220c
220d
220e
220f
354a
356a
354b
354c
354d
126
124a
124b

FIG. 13A

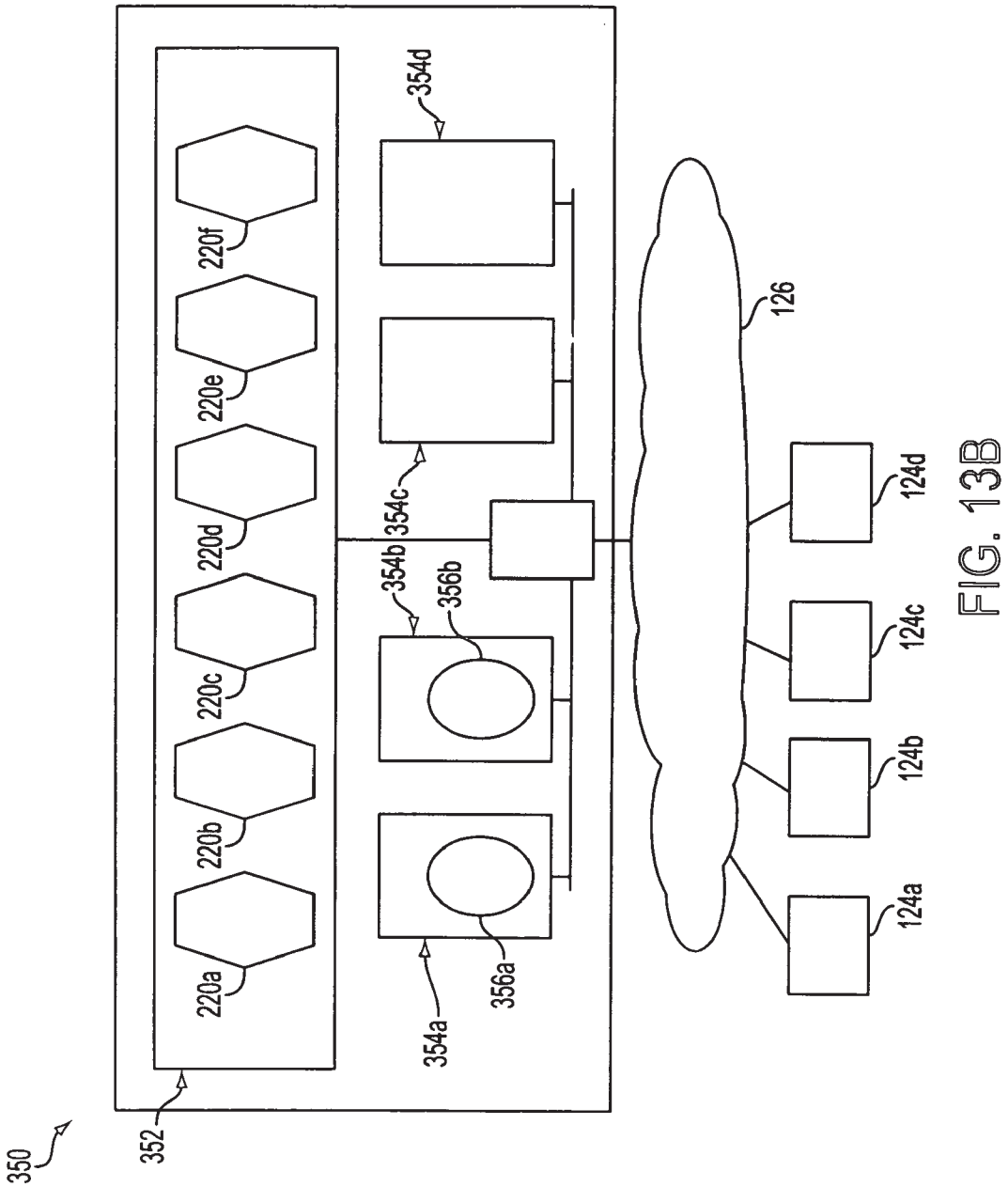

350
352
220a
220b
220c
220d
220e
220f
354a
356a
354b
356b
354c
354d
126
124a
124b
124c
124d

FIG. 13B

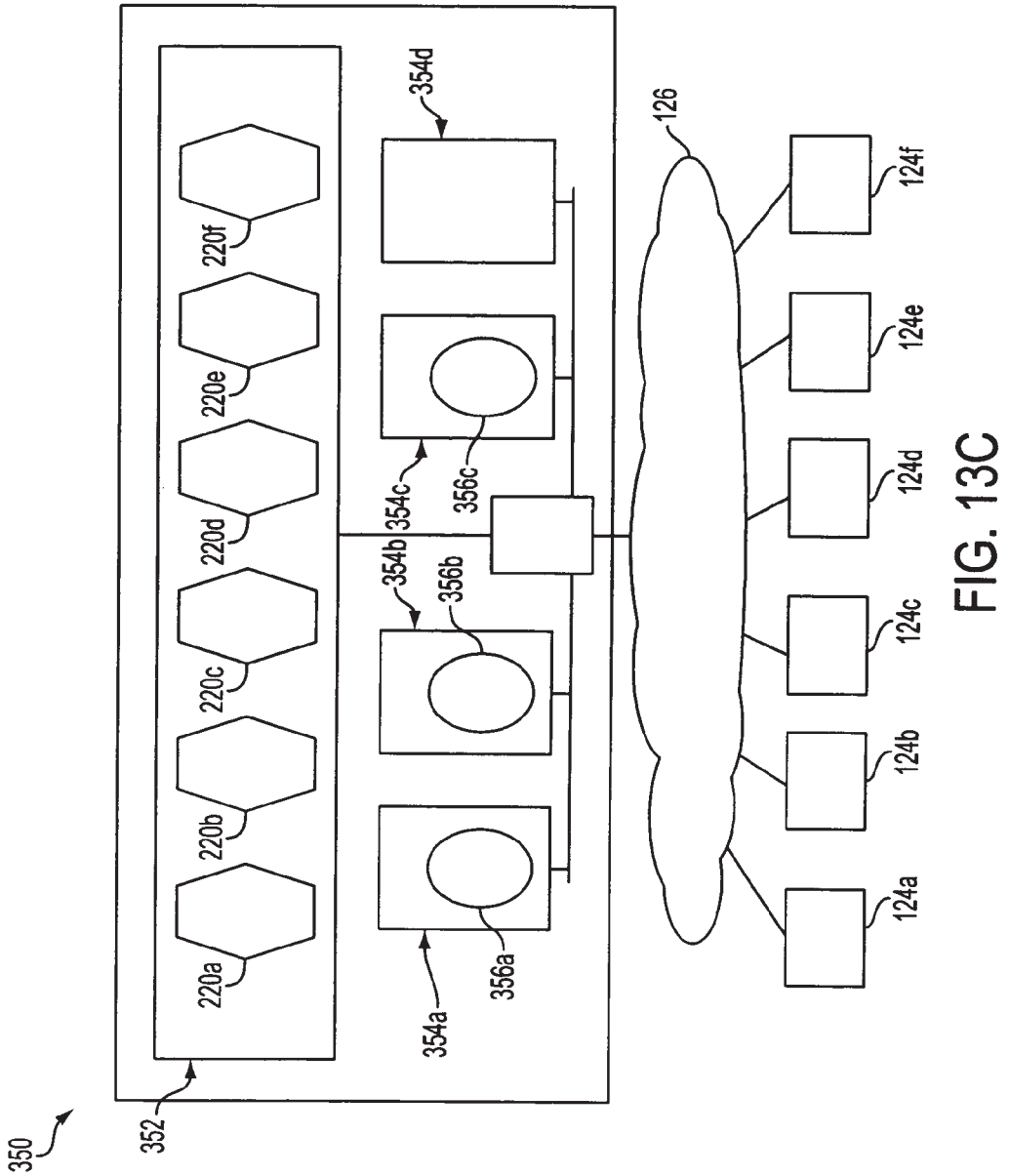

350
352
220a
220b
220c
220d
220e
220f
354a
356a
354b
356b
354c
356c
354d
126
124a
124b
124c
124d
124e
124f

FIG. 13C

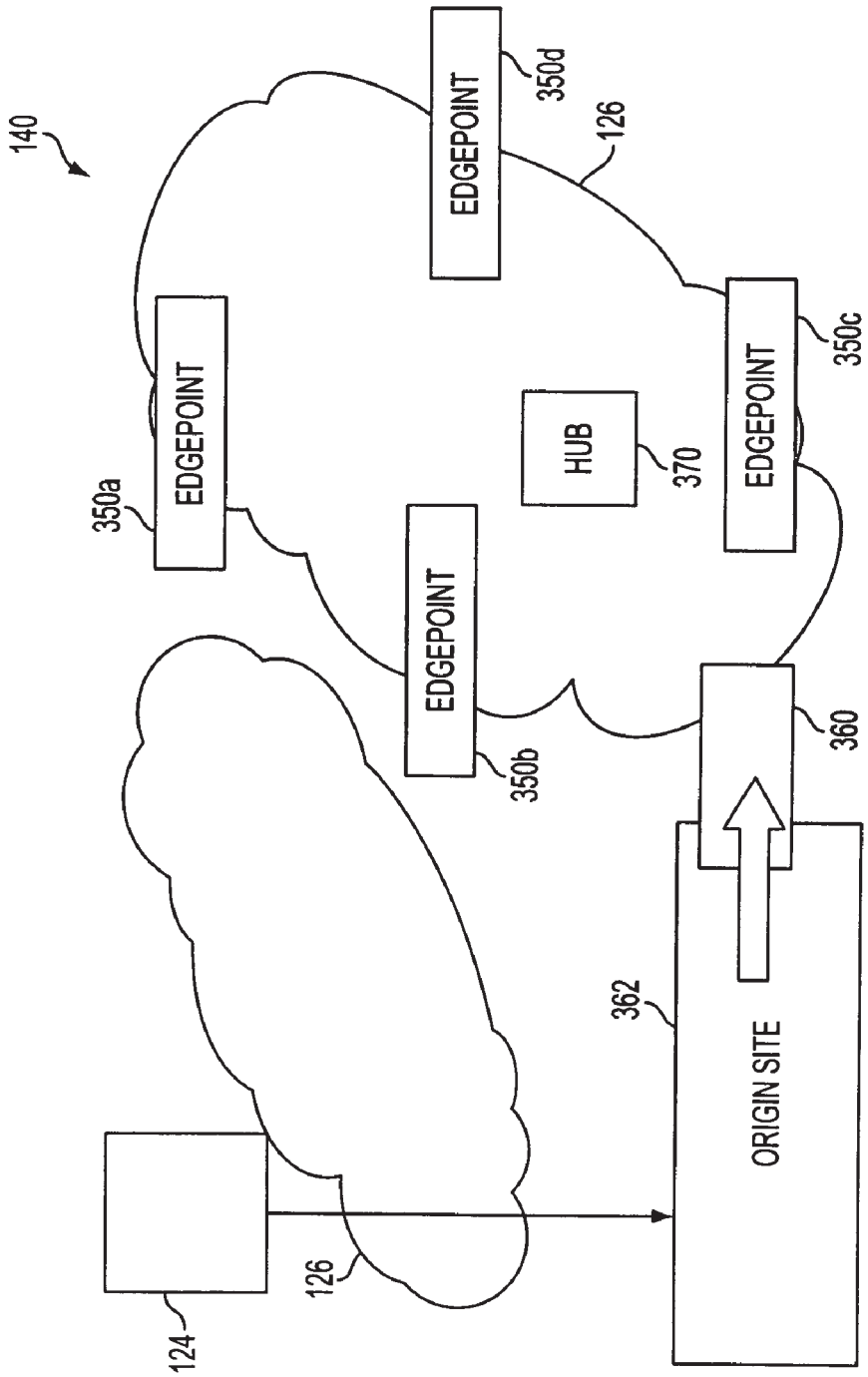

140
EDGEPOINT
350d
126
EDGEPOINT
350a
HUB
370
EDGEPOINT
350c
EDGEPOINT
350b
360
362
ORIGIN SITE
126
124

FIG. 14A

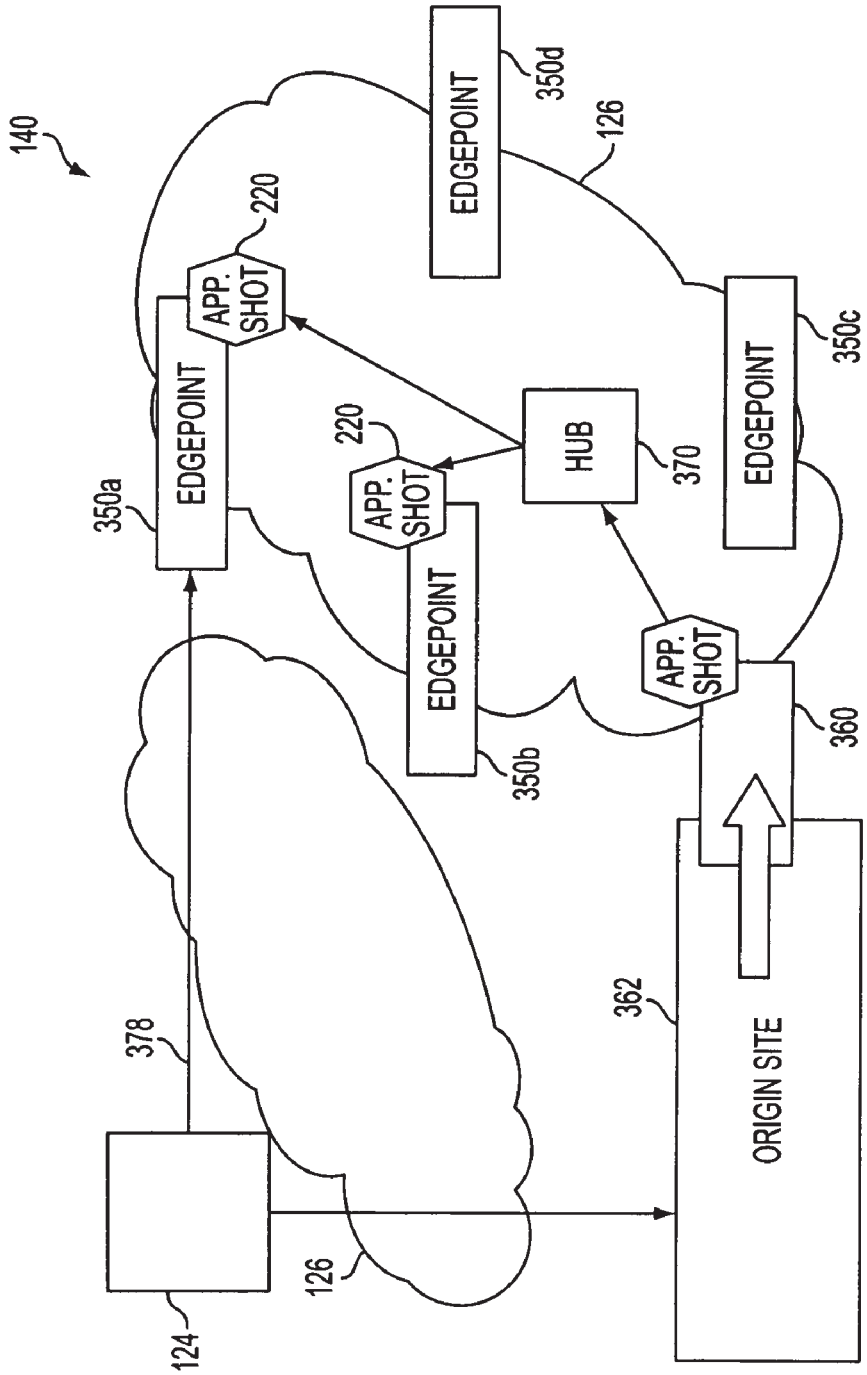

140
EDGEPOINT
350d
126
220
APP. SHOT
EDGEPOINT
350a
220
APP. SHOT
HUB
370
EDGEPOINT
350c
EDGEPOINT
350b
APP. SHOT
360
362
ORIGIN SITE
378
126
124

FIG. 14B

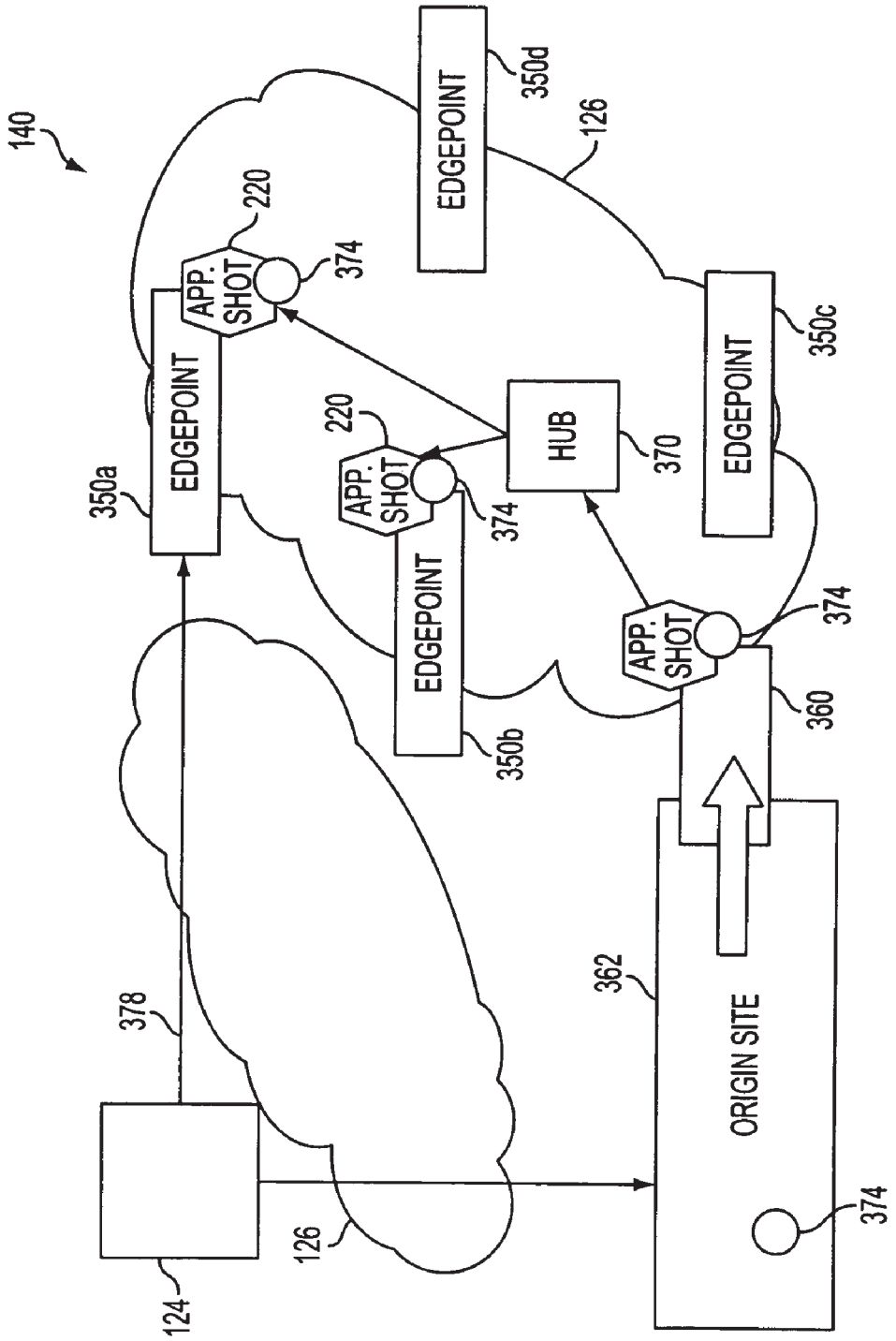

140
EDGEPOINT
350d
126
220
APP. SHOT
374
EDGEPOINT
350a
220
APP. SHOT
374
HUB
370
EDGEPOINT
350c
EDGEPOINT
350b
APP. SHOT
374
360
362
ORIGIN SITE
374
378
126
124

FIG. 14C

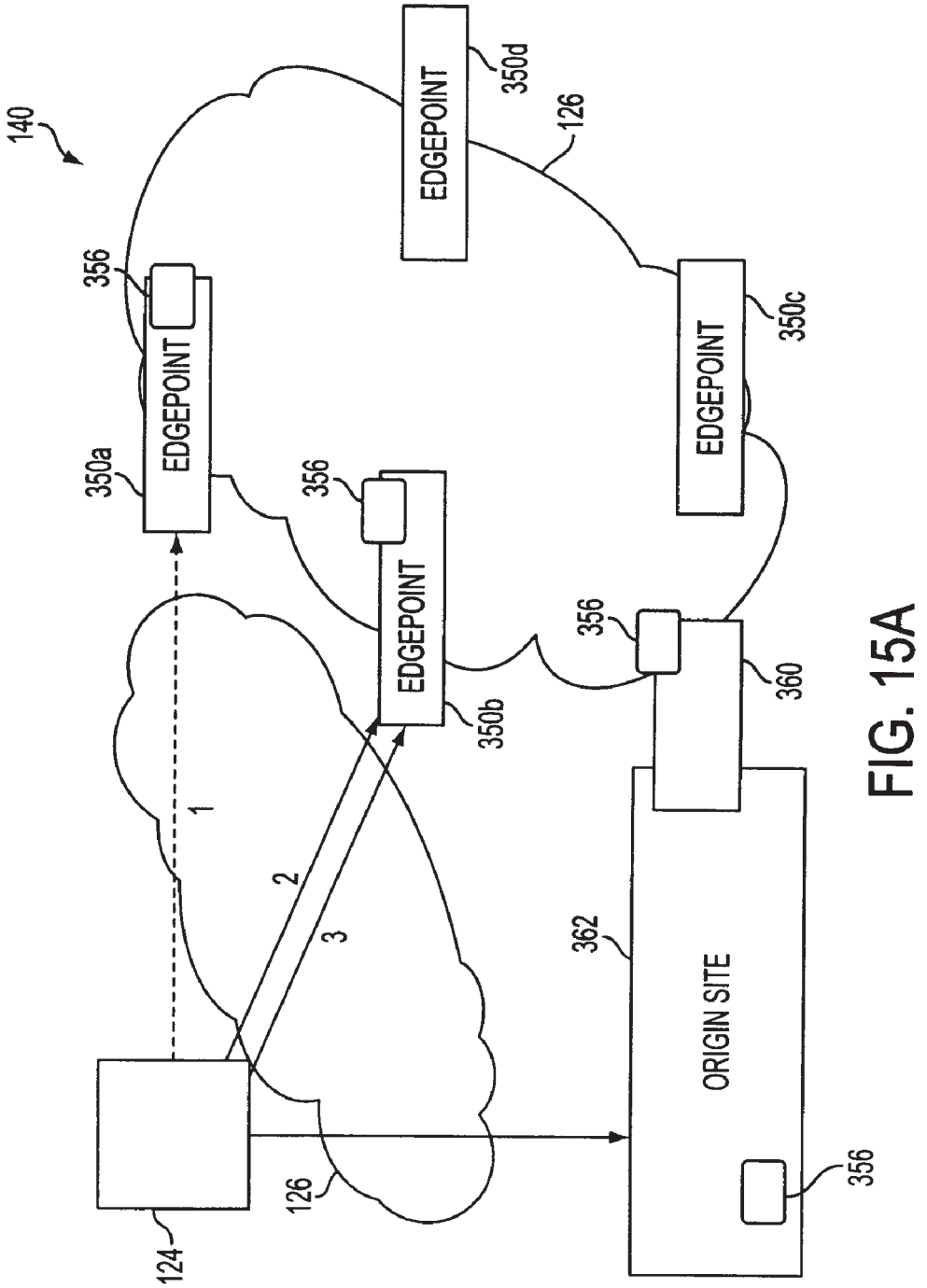

140
EDGEPOINT
350d
126
356
EDGEPOINT
350a
EDGEPOINT
350c
356
EDGEPOINT
350b
356
360
1
2
3
362
ORIGIN SITE
356
126
124

FIG. 15A

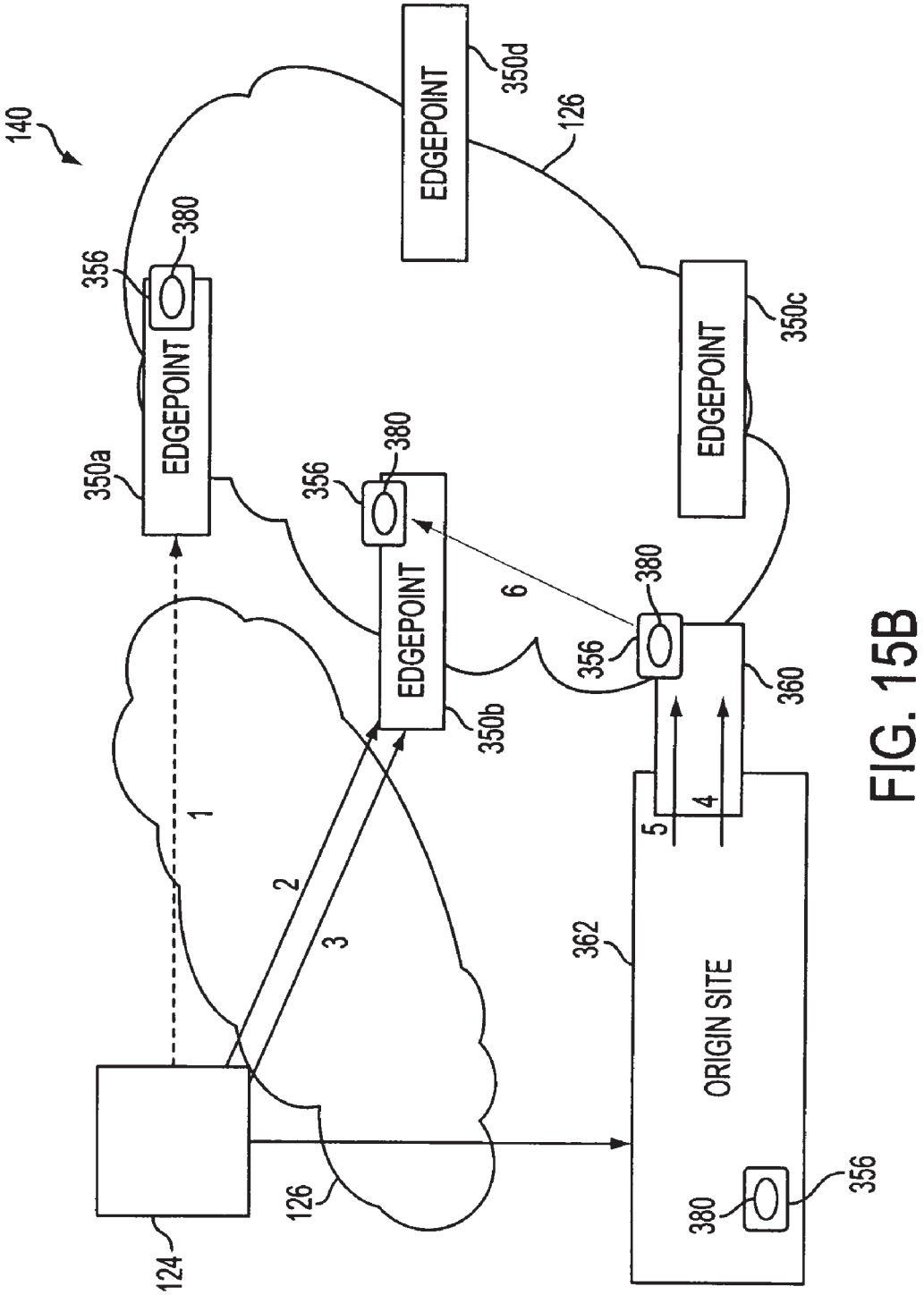

140
EDGEPOINT
350d
126
356
380
EDGEPOINT
350a
EDGEPOINT
350c
356
380
EDGEPOINT
350b
6
356
380
360
1
2
3
4
5
362
ORIGIN SITE
380
356
126
124

FIG. 15B

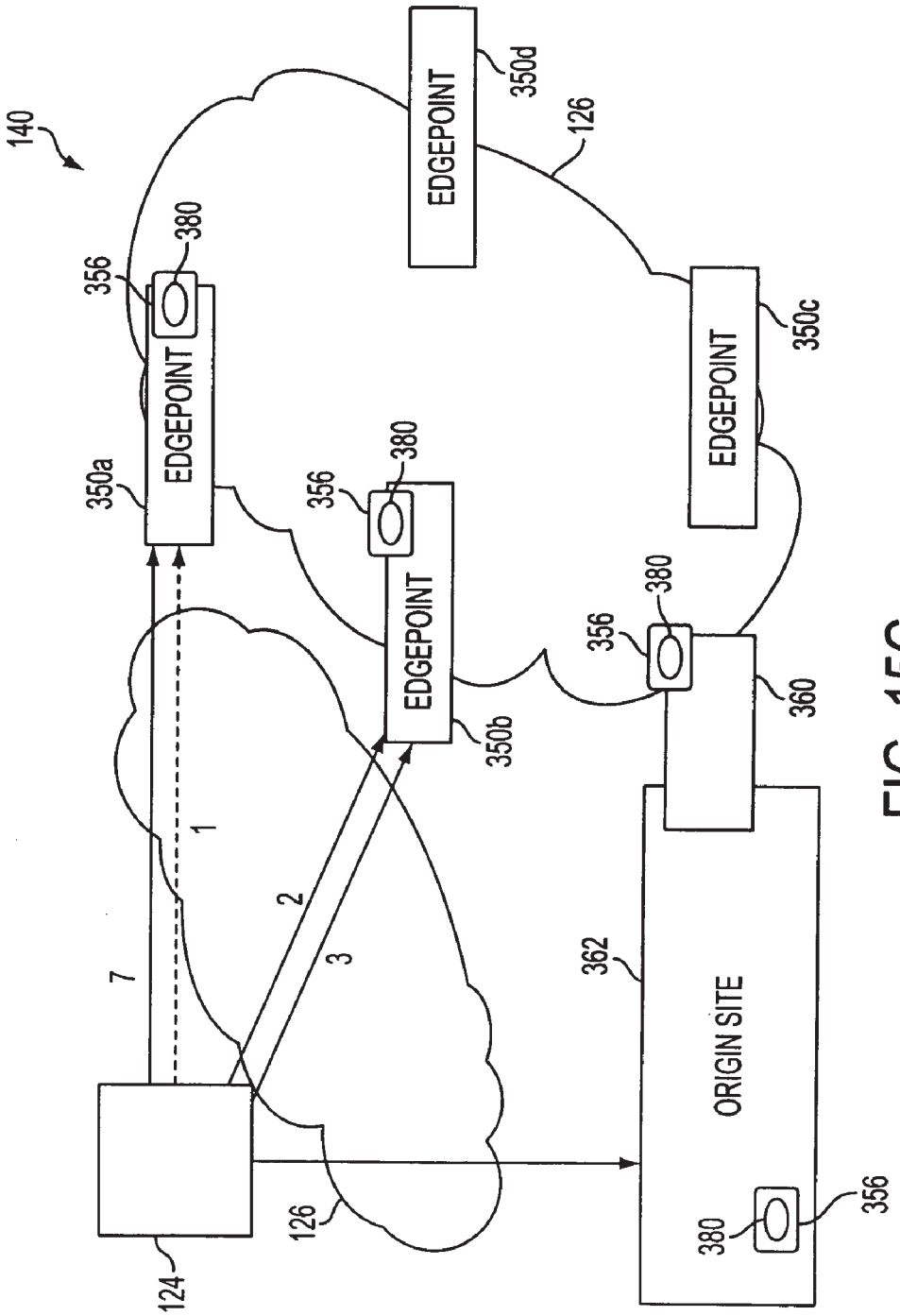

140
EDGEPOINT
350d
126
EDGEPOINT
350c
EDGEPOINT
350a
EDGEPOINT
350b
356
380
360
362
ORIGIN SITE
124
126
1
2
3
7

FIG. 15C

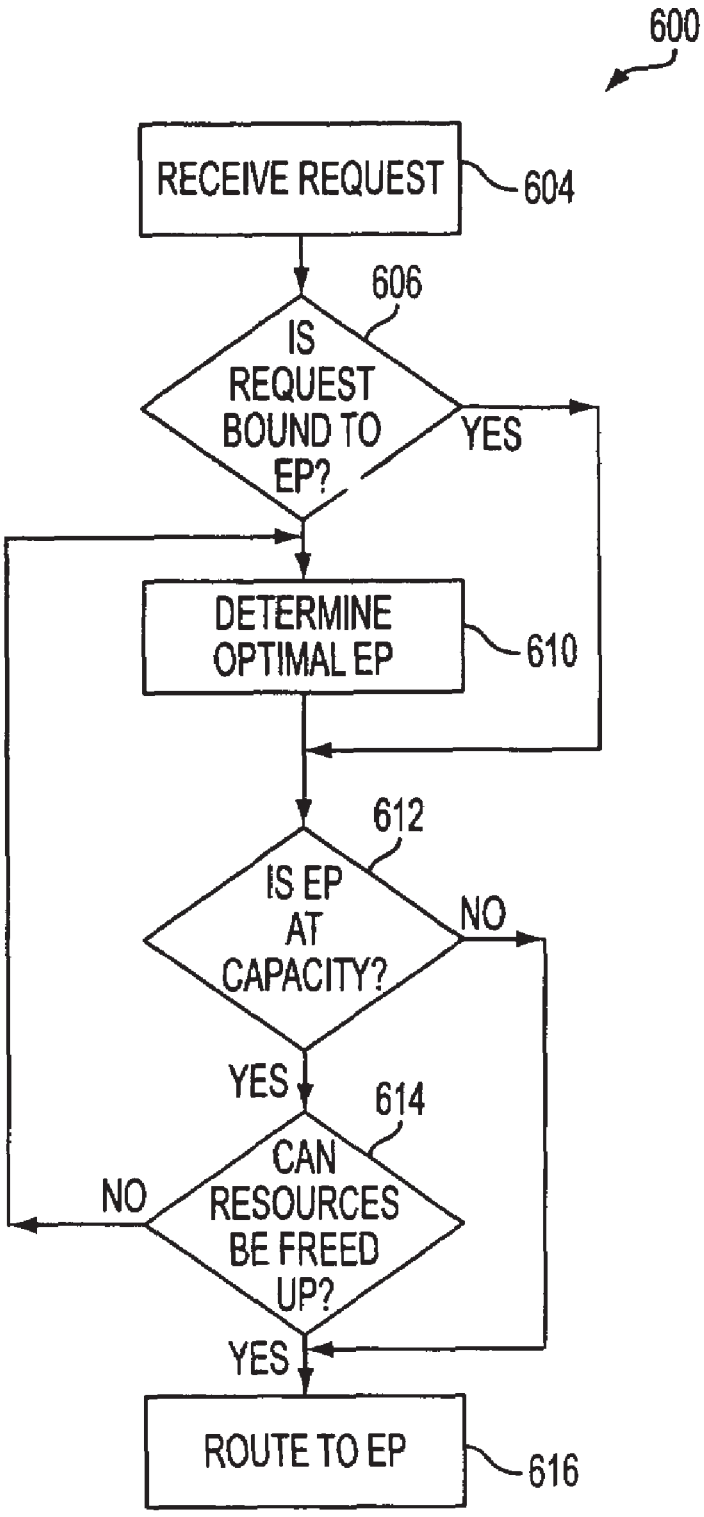

600
RECEIVE REQUEST
604
606
IS REQUEST BOUND TO EP?
YES
DETERMINE OPTIMAL EP
610
612
IS EP AT CAPACITY?
NO
YES
614
CAN RESOURCES BE FREED UP?
NO
YES
ROUTE TO EP
616

FIG. 16

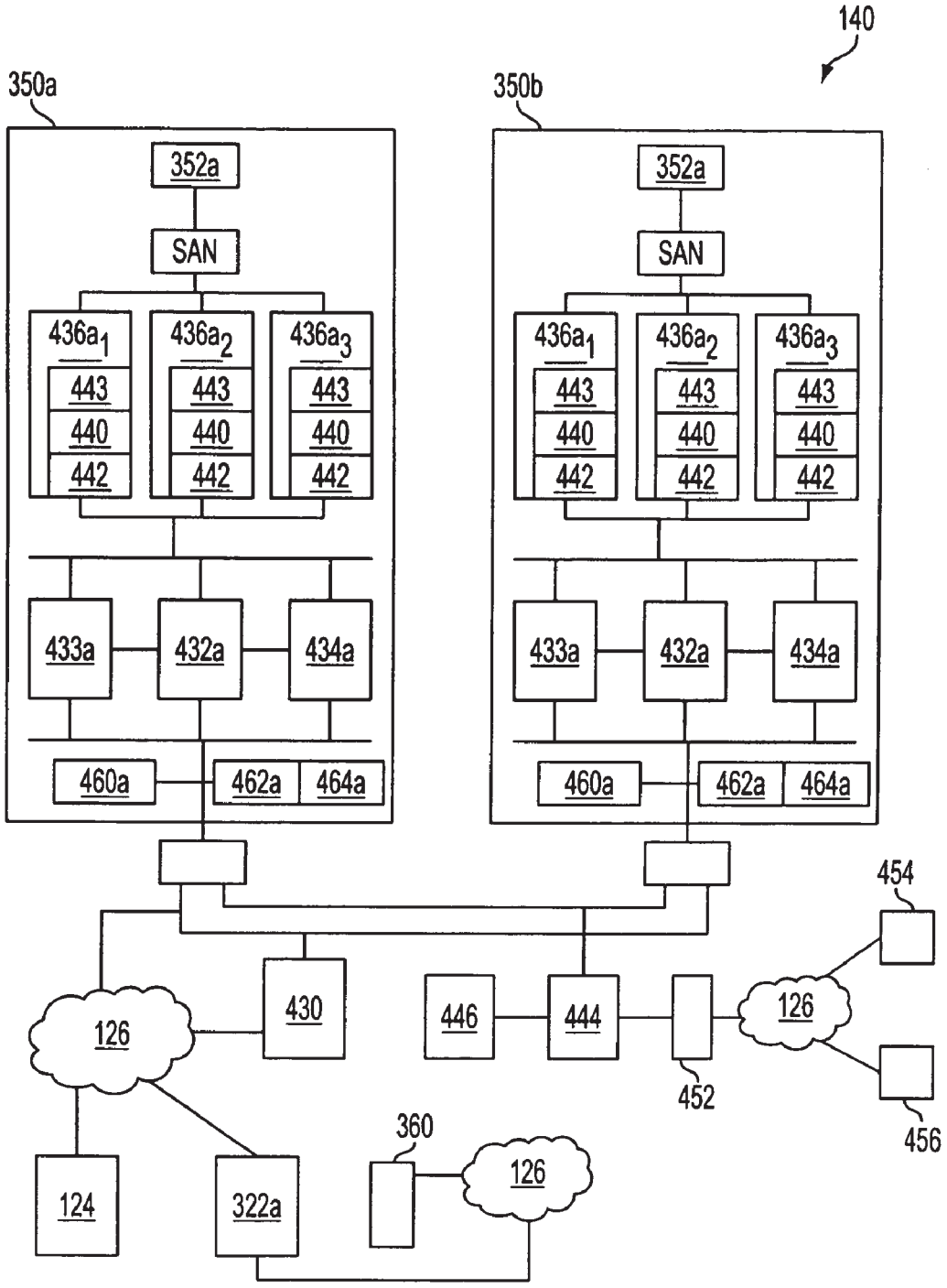

140
350a
350b
352a
SAN
436a1
436a2
436a3
443
440
442
433a
432a
434a
460a
462a
464a
454
126
430
446
444
452
456
360
124
322a

FIG. 17

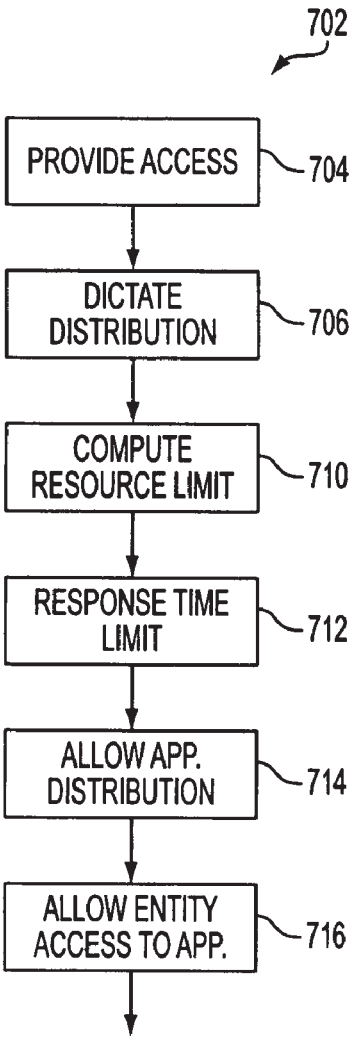

702
PROVIDE ACCESS
704
DICTATE DISTRIBUTION
706
COMPUTE RESOURCE LIMIT
710
RESPONSE TIME LIMIT
712
ALLOW APP. DISTRIBUTION
714
ALLOW ENTITY ACCESS TO APP.
716

FIG. 18

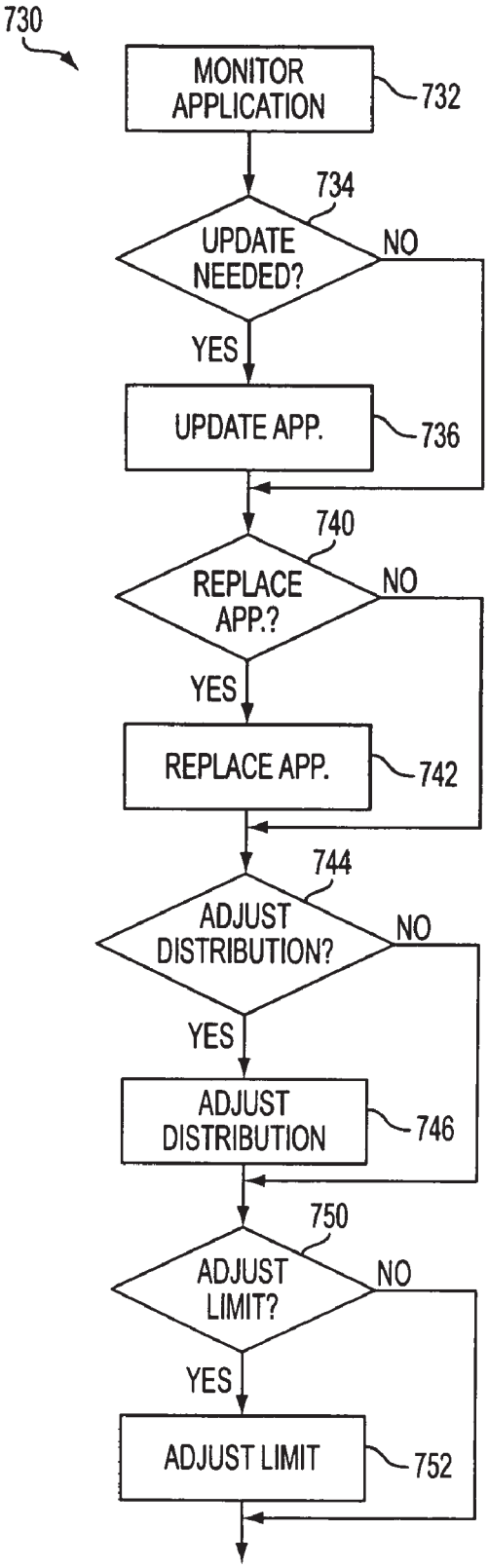

730
MONITOR APPLICATION
732
734
UPDATE NEEDED?
NO
YES
UPDATE APP.
736
740
REPLACE APP.?
NO
YES
REPLACE APP.
742
744
ADJUST DISTRIBUTION?
NO
YES
ADJUST DISTRIBUTION
746
750
ADJUST LIMIT?
NO
YES
ADJUST LIMIT
752

FIG. 19

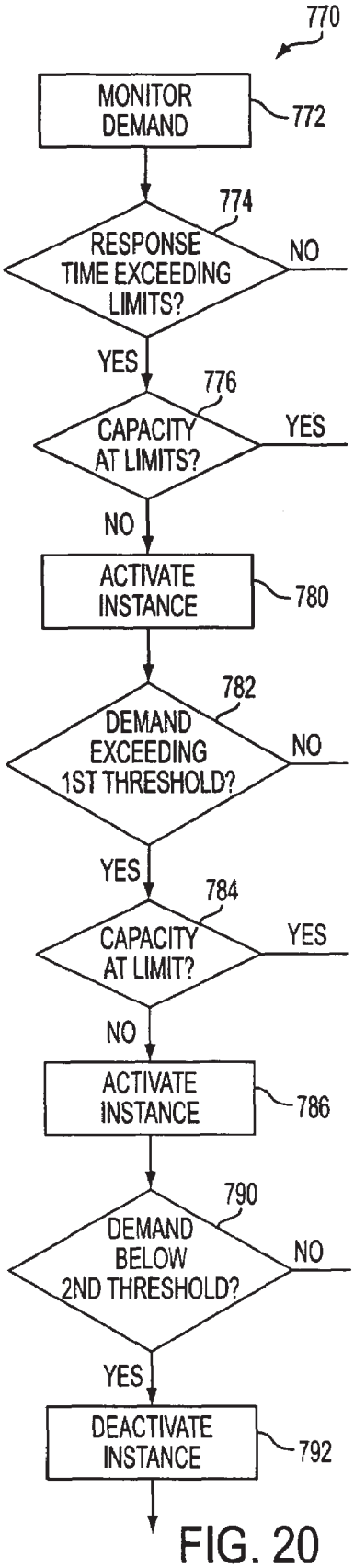

770
MONITOR DEMAND
772
774
RESPONSE TIME EXCEEDING LIMITS?
NO
YES
776
CAPACITY AT LIMITS?
YES
NO
ACTIVATE INSTANCE
780
782
DEMAND EXCEEDING 1ST THRESHOLD?
NO
YES
784
CAPACITY AT LIMIT?
YES
NO
ACTIVATE INSTANCE
786
790
DEMAND BELOW 2ND THRESHOLD?
NO
YES
DEACTIVATE INSTANCE
792

FIG. 20

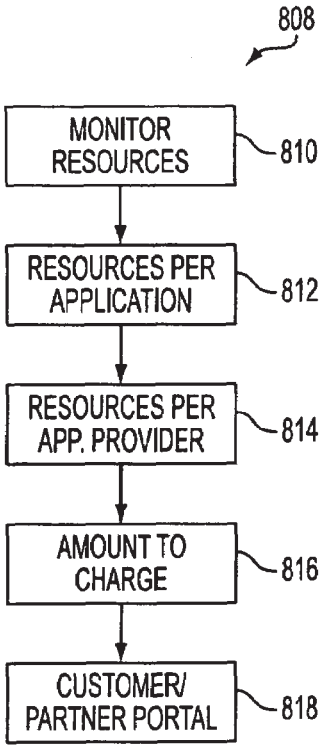

808
MONITOR RESOURCES
810
RESOURCES PER APPLICATION
812
RESOURCES PER APP. PROVIDER
814
AMOUNT TO CHARGE
816
CUSTOMER/ PARTNER PORTAL
818

FIG. 21

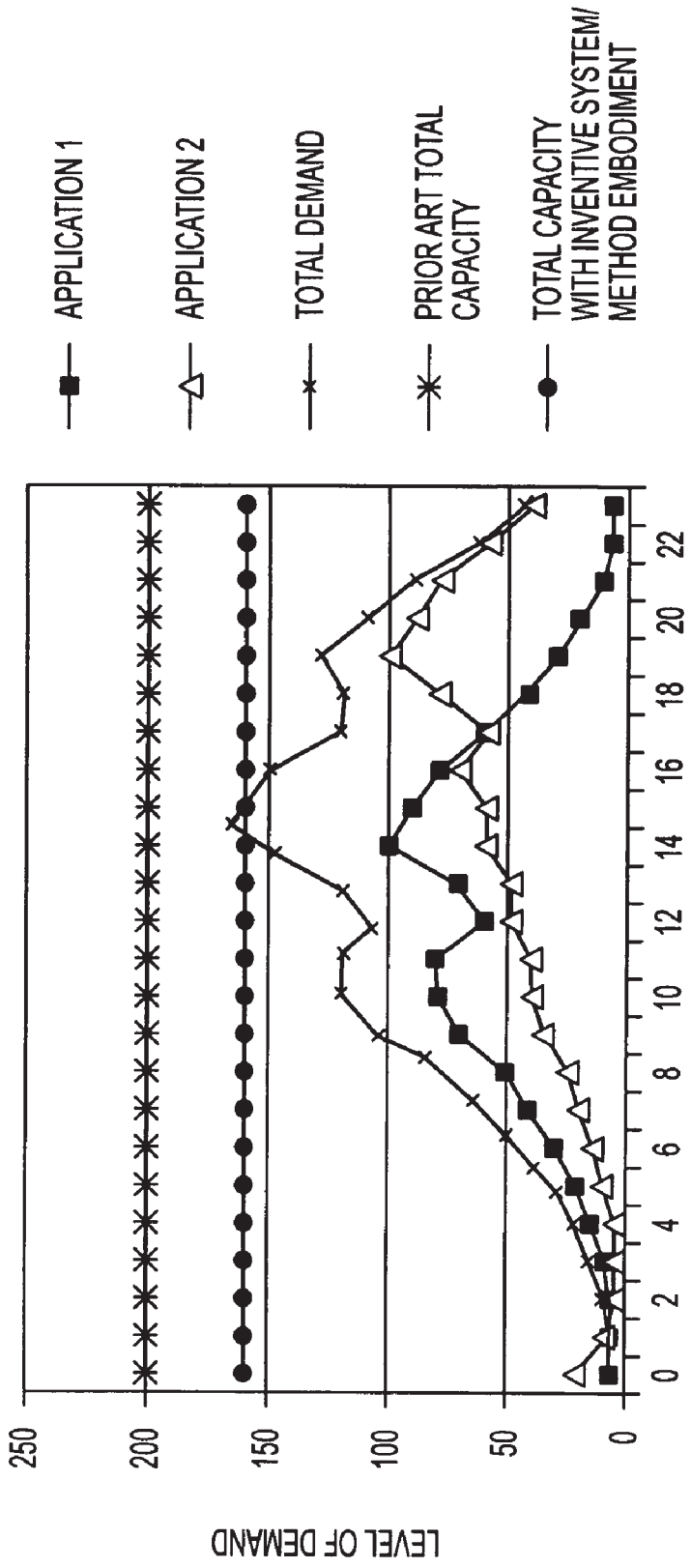

APPLICATION 1
APPLICATION 2
TOTAL DEMAND
PRIOR ART TOTAL CAPACITY
TOTAL CAPACITY WITH INVENTIVE SYSTEM/ METHOD EMBODIMENT
LEVEL OF DEMAND
250
200
150
100
50
0
0
2
4
6
8
10
12
14
16
18
20
22
TIME OF DAY

FIG. 22

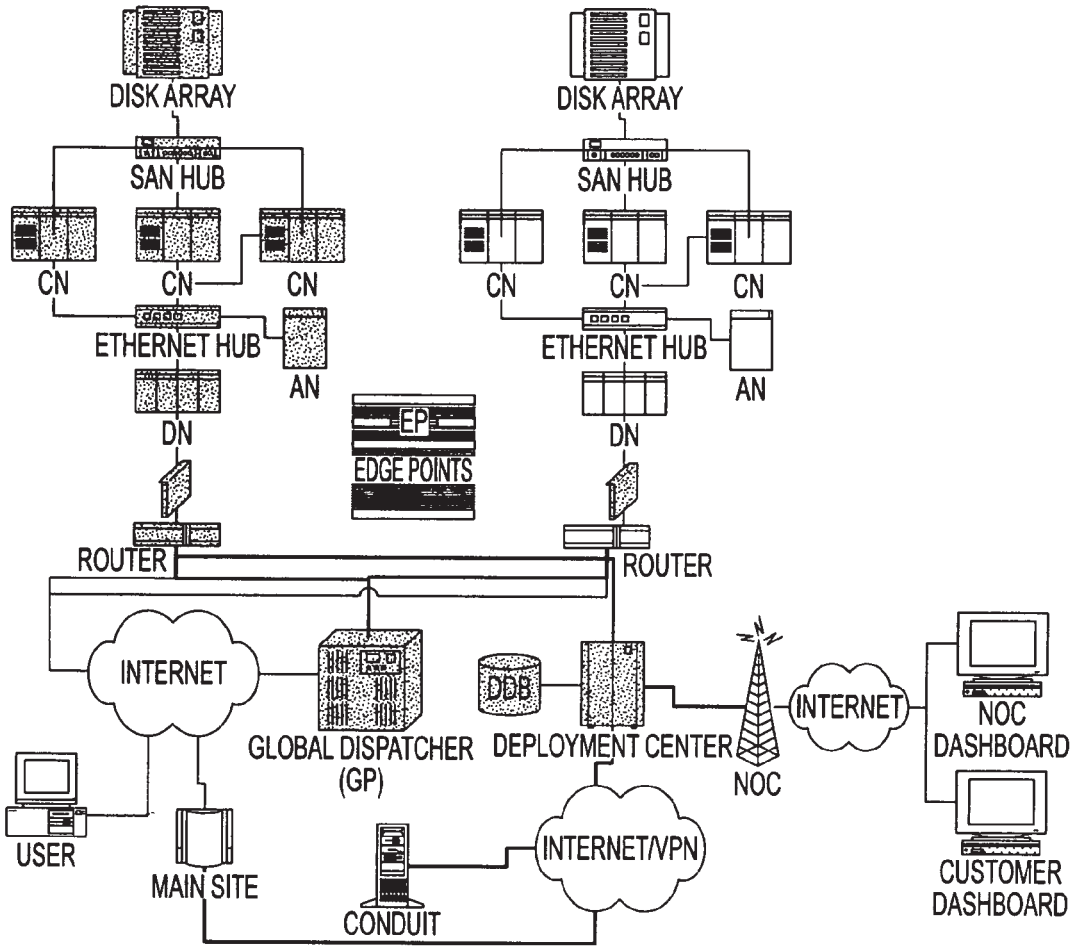

DISK ARRAY
DISK ARRAY
SAN HUB
SAN HUB
CN
CN
CN
CN
CN
CN
ETHERNET HUB
ETHERNET HUB
AN
AN
DN
DN
EP
EDGE POINTS
ROUTER
ROUTER
INTERNET
GLOBAL DISPATCHER
(GP)
DDB
DEPLOYMENT CENTER
NOC
INTERNET
NOC
DASHBOARD
USER
MAIN SITE
CONDUIT
INTERNET/VPN
CUSTOMER
DASHBOARD

FIG. 23

METHOD SYSTEM AND APPARATUS FOR PROVIDING PAY-PER-USE DISTRIBUTED COMPUTING RESOURCES

RELATED APPLICATIONS

The present application claims priority to and incorporates the following applications in their entirety by reference:

A Method and Apparatus for Providing Pay-Per-Use, Distributed Computing Capacity, U.S. Provisional Application Serial No. 60/232,052, filed on Sep. 12, 2000;

Snapshot Virtual Templating, U.S. patent application Ser. No. 09/684,373, filed on Oct. 5, 2000;

Dynamic Symbolic Link Resolution, U.S. patent application Ser. No. 09/680,560, filed on Oct. 5, 2000;

Snapshot Restore of Application Chains and Applications, U.S. patent application Ser. No. 09/680,847, filed on Oct. 5, 2000;

Virtual Resource-ID Mapping, patent application Ser. No. 09/680,563, filed on Oct. 5, 2000; and

Virtual Port Multiplexing, patent application Ser. No. 09/684,457, filed on Oct. 5, 2000.

FIELD OF INVENTION

In general the invention pertains to computer application processing, more particularly to distributed computing for computer application processing, and most particularly to system and method for providing computer application processing with dynamic capacity control and pay-per-use usage charging on an on-demand basis.

BACKGROUND

There is a trend of emerging computing infrastructure aimed at on-demand services, particularly for Internet or other distributed networked computing services. There are basically three categories of on-demand services that are currently available. The first is content delivery, the second is storage, and the third is bandwidth. These services are provided as needed or on-demand, based on a user's needs at any given time. For example, if a first data provider needs greater storage space, an on-demand storage provider simply allocates a greater amount of storage memory to that user, and the first data provider is charged based on the amount of memory space used. If the first data provider no longer needs that amount of memory and deletes information, the on-demand storage provider is then able to re-allocate that memory space to an alternative data provider and the first data provider is charged less because of the reduced storage use.

One of the problems that companies with substantial IT investments face is that it is very difficult for them to predict how much demand they will have for their applications (capacity planning). Therefore, it is extremely difficult for them to determine how large a server farm to deploy which will allow greater user access to their services.

Another problem faced by application or website providers is the continued need for resource capacity to provide adequate service to their users. This is also referred to as the scalability problem. FIG. **1** shows a simplified block diagram representation of the diseconomy of scale resulting in the server infrastructure. What is seen is that application providers are in what is sometimes referred to as a high growth spiral. In the high growth spiral the application provider starts by building a service **52** to gain application users or customers **54**. The increase in users results in an increase in the application providers server loads **56**. This increased server load causes an increase in response time and often results in the application provider's sites failing or going down, which may result in a loss **60** of users. The application provider must then invest in more resources and infrastructure **62** to reduce response time, improve reliability and keep their users happy **64**. This increased response time, and reliability then attracts more users **54**, which returns the application provider back to a point where the increased load demands stress or tax the application provider's servers **56**, resulting again in a slower response time and a decrease in reliability. Thus, application providers are constantly going around in this high growth spiral.

FIG. **2** shows a graphical representation of the cost peruser to increase resource capacity. One of the problems faced by application providers is that the cost of server infrastructure may typically increase faster than the current number of users so that costs are non-linear. This means that as the application provider's server farm gets more complex the cost delta **70** to add enough capacity to service one additional user increases. Thus, the cost **70** of continuing to grow increases dramatically in relation to the cost per user. With most every other business, as the business grows, economies of scale come into effect and the costs per user served actually decreases **72**. This is one of the real problems faced by application providers.

Bottlenecks exist in various system resources, such as memory, disk I/O, processors and bandwidth. To scale infrastructure to handle higher levels of load requires increased levels of these resources, which in turn require space, power, management and monitoring systems, as well as people to maintain and operate the systems. As user load increases, so does complexity, leading to costs increasing at a faster rate than volume.

Another problem with providing application processing or services is the amount of capacity that will be needed at start-up, as well as the capacity needs in the future to maintain response time and reliability. These are both start-up costs. It is relatively impossible to predict in advance, with any degree of accuracy, just how successful a site or service is going to be prior to launching and activating the site.

FIG. **3** shows a graphical representation of user capacity demands of an application provider. When an application provider installs a certain number of servers, whatever that number is, the provider has basically created a fixed capacity **74**, while demand itself may be unpredictable. Because of the unpredictability of usage demands on servers, that fixed capacity **74** will be either too high **76**, and the application provider did not have as many users as anticipated resulting in wasted capacity **76** and wasted capital. Or the fixed capacity **74** was too low **80**, and the application provider obtained more users than predicted, resulting in insufficient capacity **80**. Thus, if the fixed capacity **74** is too high, the application provider has invested too much capital **76**. If the fixed capacity **74** is too low **80**, the application provider has users who are dissatisfied because the user does not get the service they need or it takes too long to get responses. This unpredictability is an extremely difficult problem faced by companies providing application processing and services and is particularly severe for those providing services over the Internet simply because of the dynamics of the Internet. The demand is completely unpredictable, and is substantially impossible to plan.

One problem faced by on-line application providers or other users of distributed computing networks is that the network is actually very slow for interactive services as a result of large traverses across the network, because communication signals run into the inherent latency of the network. For example, if an Internet user is in New York, but that New York user want to access a website that is serviced in Los

Angeles, the New York user must be routed or hopped all the way across the U.S. Sometimes users will be routed all the way around the world, to get to a specific site. These long distance routings run into large amounts of latency delay. This inherent latency of distributed networks is amplified by the significant increase in the number of interactive services deployed by application and website providers having very active pages or sites. Further, there is a general trend towards customized pages per user. These are sites which are custom created by the server or application for a particular user. These customized sites reduce caching effects to substantially zero. Thus, a customized page, created for that specific user, is generated at the server origin site and routed all the way back across the net to the user adding further inherent delays in the response time. This adds up to a very slow service for more complex interactive services.

In prior art systems, application providers wishing to provide applications have to buy or lease a server, then they must buy or develop the applications that are going to be loaded and run on that server, load the server, and activate the server to provide access to that application. The server is a fully dedicated resource, so that 100% of the time an application is dedicated to a specific server.

Prior art application processing systems require an application provider to route a user to a single central site to allow access to the applications. Every user attempting to access the application is directed to the single central site. Thus, resulting in a bottle neck at the central site. In the prior art single server or single central site, the application provider, however, does maintain access to and control over the application. In some systems where the application provider outsources their server capacity, the application provider must select from a preselected limited number of applications. Further, the application provider no longer has direct control over the application. Any changes desired require the application provider to submit a request to the server provider. Then the server provider must schedule a time at low demands to take the server down to make the changes. This process results in large lag times between the decision to make changes and the implementation of those changes.

SUMMARY

The novel method, apparatus, computer readable medium and computer program product of the present invention provides on-demand, scalable computational resources to application providers over a distributed network and system. The resources are made available upon receiving requests for a first application. Once a request is received, routing of the request is determined and the request is routed to access the first application. The application provider is then charged based on the amount of resources utilized to satisfy the request. In determining routing the method and apparatus determines if a first instance of a first application is active, and if the first instance is at a capacity. A first set of compute resources is provided to satisfy the first request and the amount charged to the first application provider is increased based on the first set of compute resources. In one embodiment, the method and apparatus activates a second instance of the first application on a second set of the available compute resources if the first instance is at capacity and the amount charged to the first application provider is increased based on the second set of compute resources. As a result, resources needed are dynamically available on demand, and freed when not needed. The application provider is only charged for services that are actually used.

In one embodiment, a third set of compute resources are freed up if the compute resources are not available. A second instance of the first application is restored on a fourth set of compute resources such that the fourth set of compute resources includes at least a portion of the freed up third set of compute resources, and the amount charged to the first application provider is increased based on the fourth set of compute resources. In freeing up resources, a first instance of a second application is snapshotted, wherein the second application is provided by a second application provider, and an amount charged to the second application provider is reduced based on the freed up third set of compute resources.

The method and apparatus provides application providers with access to the network, where the network includes the distributed compute resources configured to provide the application processing and allows the application providers to distribute applications onto the network to utilize the distributed compute resources for processing of the applications. The application providers are further capable of monitoring, updating and replacing the distributed applications. The method and apparatus increases the amount of compute resources utilized in providing processing for an application as demand for the application increases. As the amount of compute resources is increased the amount charged to the application provider is increased based on the amount of compute resources utilized. As demand for the application falls, the amount of resources is reduced and the amount charged the application provider is reduced.

In one embodiment, the apparatus for providing the on-demand compute resources includes a first resource manager, at least one snapd (snapshot or snapshot daemon) module configured to snapshot an active application, at least one restored (restore daemon) module configured to restore a snapshotted application, and a first set of compute resources configured to provide application processing. The resource manager couples with and provide at least some control to the snapd module, restored module and the first set of compute resources. The resource manager is further configured to monitor the amount of the first set of compute resources utilized in providing application processing. In one embodiment, the apparatus includes at least one perfd (performance or performance daemon) module coupled with the first resource manager and the first set of compute resources, and is configured to monitor the first set of computational resources and provide the resource manager with compute resource utilization. In one embodiment, a deploy module couples with the first resource manager and the first set of compute resources, and is configured to receive at least one application from at least one of the application providers, and provision the first set of compute resources to be utilized in processing the at least one application. A conduit couples with the deploy module, and is configured to provide the application providers with access to the deploy module to distribute applications or updates for application processing. A local dispatcher couples with the first resource manager and the first set of compute resources, and is configured to receive directions from the resource manager and to provide routing of requests for the at least one application to the first set of compute resources. In one embodiment, the resource manager, snapd module, restored module, perfd module, local dispatch module and deploy module are cooperated into a single edgepoint. In one embodiment, the apparatus includes a plurality of edgepoints distributed to provide the on-demand, distributed compute resources.

In one embodiment, the apparatus includes a plurality of sets of compute resources and a plurality of resource managers, such that the sets of compute resources are utilized for

application processing. Further, a global dispatcher coupled with the plurality of resource managers, wherein the global dispatcher is configured to receive requests for at least one application and to route the requests to an optimal resource manager. In one embodiment, the apparatus includes one or more compute modules which comprise at least a snapd module, a restored module and at least a third set of compute resources.

In one embodiment the novel network providing on-demand compute resources includes a first means for application processing configured to provide application processing, a first application distributed onto the network and configured to be processed by the first means for application processing, a first means for managing application processing coupled with the first means for application processing, and configured to activate at least a first instance of the first application on a first set of the first means for application processing based on a first amount of demand for the first application. The network further includes a means for monitoring coupled with the first means for application processing, and configured to monitor at least the first set of the first means for application processing utilized to provide the entity with access to the first instances of the first application, and a means for determining an amount to charge coupled with the first means for application processing, and configured to determine an amount to be charged based on the first set of the first means for application processing utilized in providing the entity with access to the first instance of the first application. The means for managing application processing is further configured to activate a second instance of the first application on a second set of the first means for application processing based on a second amount of demand for the first application. The means for monitoring is further configured to monitor the second set of the first means for application processing utilized to satisfy the second amount of demand for the first application, and the means for determining an amount to charge is configured to determine an amount to be charged based on the second set of the first means for application processing utilized in providing access to the second instance of the first application. The means for managing application processing is further capable of deactivating one of the first and second instances of the first application based on a third amount of demand for the first application In one embodiment, the method and apparatus includes a plurality of means for application processing, and a means for dispatching coupled with the plurality of means for application processing. The means for dispatching is configured to route at least one entity to an optimal means for application processing allowing the at least one entity access to at least one application. In one embodiment means for application processing is an edgepoint. In one embodiment, the means for dispatching is a global dispatcher. In one embodiment, the means for application processing is a compute module.

In one embodiment, the system, method, and business operating model provide a computer application processing capacity as a pay-per-use utility on demand.

BRIEF DESCRIPTION OF THE FIGURES

The invention, together with further advantages thereof, may best be understood by reference to the following description taken in conjunction with the accompanying drawings in which:

FIG. **1** shows a simplified block diagram representation of the diseconomy of scale resulting from the server infrastructure;

FIG. **2** shows a graphical representation of the cost peruser to increase resource capacity;

FIG. **3** shows a graphical representation of user capacity demands of an application provider;

FIG. **4** shows a graphical representation of the on-demand response of the present on-demand system;

FIG. **5** depicts a simplified block diagram of a business operating over the Internet, sometimes referred to as an e-business;

FIG. **6** depicts a simplified schematic block diagram of one embodiment of the novel distributed on-demand application processing system which substantially eliminates the bottleneck and tornado effects seen in the prior art;

FIG. **7** illustrates in high level block diagram form one implementation of one embodiment of the overall structure of the present invention as used in connection with a computer network such as the internet;

FIG. **8** depicts a block diagram of one embodiment of a computer for implementing the on-demand method and apparatus of the present invention in a computer readable medium;

FIG. **9** shows a simplified block diagram of one embodiment of an overall system architecture for the distributed, on-demand application processing service and system of the present invention;

FIG. **10** shows a simplified block diagram of one embodiment of the application switching architecture;

FIG. **11** depicts a simplified flow diagram of one implementation of a sequence of steps executed by the present invention to perform a snapshot of a process or application instance;

FIG. **12** illustrates a simplified flow diagram of one implementation of the sequence of steps executed to restore a snapshotted application;

FIGS. **13**A-C shows a simplified block diagram of one embodiment of an edgepoint of the present invention;

FIGS. **14**A-C show simplified block diagrams of embodiments of the present on-demand application processing system in cooperation with the preexisting internet infrastructure;

FIGS. **15**A-C show a simplified block diagram of one implementation of one embodiment of the novel on-demand apparatus and the optimal user and entity routing provided by the present invention;

FIG. **16** shows a simplified flow diagram of one implementation of one embodiment of the method and system providing on-demand compute resources;

FIG. **17** shows a simplified block diagram of one implementation of one embodiment of a novel on-demand apparatus including a plurality of edgepoints;

FIG. **18** depicts a simplified flow diagram of a process for an application provider to access and distribute applications onto the distributed, application processing system of the present invention;

FIG. **19** depicts a simplified flow diagram of one embodiment of a process for an application provider to monitor and update applications distributed onto the system;

FIG. **20** depicts a flow diagram of one embodiment of a process for monitoring demand and determining an amount to bill an application provider;

FIG. **21** depicts a simplified flow diagram of one implementation of one embodiment of a process for determining an amount of resources utilized for an application and the amount to be charged to the application provider based on the amount of resources utilized;

FIG. **22** depicts typical exemplary demand situation for two different applications (or customers) across a twenty-four hour time period; and